



US005737419A

# United States Patent [19] Ganesan

[11] Patent Number: 5,737,419  
[45] Date of Patent: \*Apr. 7, 1998

## [54] COMPUTER SYSTEM FOR SECURING COMMUNICATIONS USING SPLIT PRIVATE KEY ASYMMETRIC CRYPTOGRAPHY

[75] Inventor: Ravi Ganesan, Arlington, Va.

[73] Assignee: Bell Atlantic Network Services, Inc., Arlington, Va.

[\*] Notice: The term of this patent shall not extend beyond the expiration date of Pat. No. 5,535,276.

[21] Appl. No.: 660,403

[22] Filed: Jun. 7, 1996

### Related U.S. Application Data

[63] Continuation-in-part of Ser. No. 338,128, Nov. 9, 1994, Pat. No. 5,535,276.

[51] Int. Cl.<sup>6</sup> ..... H04K 1/00

[52] U.S. Cl. .... 380/21; 380/30

[58] Field of Search ..... 380/21, 30, 49

### [56] References Cited

#### U.S. PATENT DOCUMENTS

4,200,770	4/1980	Hellman et al. .
4,218,582	8/1980	Hellman et al. .
4,405,829	9/1983	Rivest et al. .
4,424,414	1/1984	Hellman et al. .
4,736,423	4/1988	Matyas .
4,995,082	2/1991	Schnorr .
5,276,737	1/1994	Micali .
5,299,263	3/1994	Beller et al. .
5,375,169	12/1994	Scheidt et al. .

#### OTHER PUBLICATIONS

C. Boyd, Cryptography and Coding: "Digital Multisignatures", 15-17 Dec. 1986, pp. 241-246.

Kohl, John et al., "The Kerberos™ Network Authentication Service (V5), Internet-Draft, Sep. 1, 1992, pp. 1-69.

Bellovin, Steven M. et al., "Encrypted Key Exchange: Password-Based Protocols Secure Against Dictionary Attacks", IEE, 1992, pp. 72-84.

Schneier, B., "Applied Cryptography, Protocols, Algorithms and Source Code in C", pp. 428-436, John Wiley & Sons, NY 1994 (re Kent, S., Privacy Enhancement for Internet Electronic Mail: Part II: Certificate Based Key Management, Internet RFC 1422, Feb. 1993).

Schneier, B., "Applied Cryptography, Protocols, Algorithms and Source Code in C", p. 424, John Wiley & Sons, NY 1994 (re Kohl, J.T., The Evolution of the Kerberos Authentication Service, EurOpen Conference Proceedings, May 1991).

(List continued on next page.)

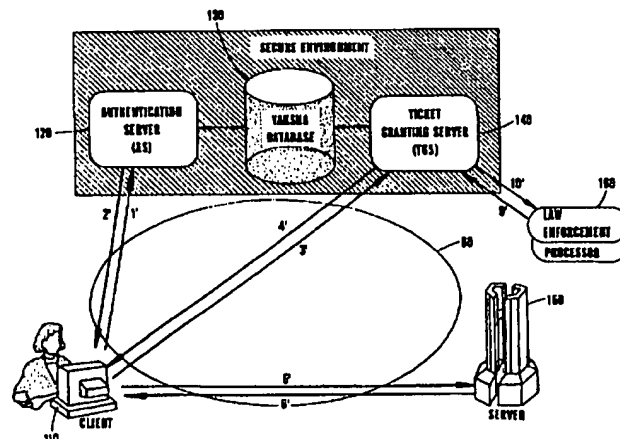
Primary Examiner—David C. Cain

Attorney, Agent, or Firm—Lowe, Price, LeBlanc & Becker

### [57] ABSTRACT

A programmed computer secures communications between users of a crypto-system in which each user has an associated asymmetric crypto-key with a public key portion accessible to all system users and a corresponding private key portion having a first private key portion known only to the associated user and a corresponding second private key portion. The computer includes a processor programmed to generate a temporary asymmetric crypto-key having a first temporary key portion and an associated second temporary key portion. The computer then encrypts the second temporary key portion with the first private key portion of a first user crypto-key associated with a first user to form a first encrypted message. The processor directs the issuance of the first encrypted message to a second user having access to the second private key portion of the first user crypto-key. The processor next applies the public key portion of the first user crypto-key to decrypt a second encrypted message generated by the second user, which includes the first encrypted message encrypted with the second private key portion of the first user crypto-key, to thereby authenticate the second user to the first user. The computer also includes a storage medium for storing the first temporary key portion, and the public key portion of the first user crypto-key.

27 Claims, 9 Drawing Sheets

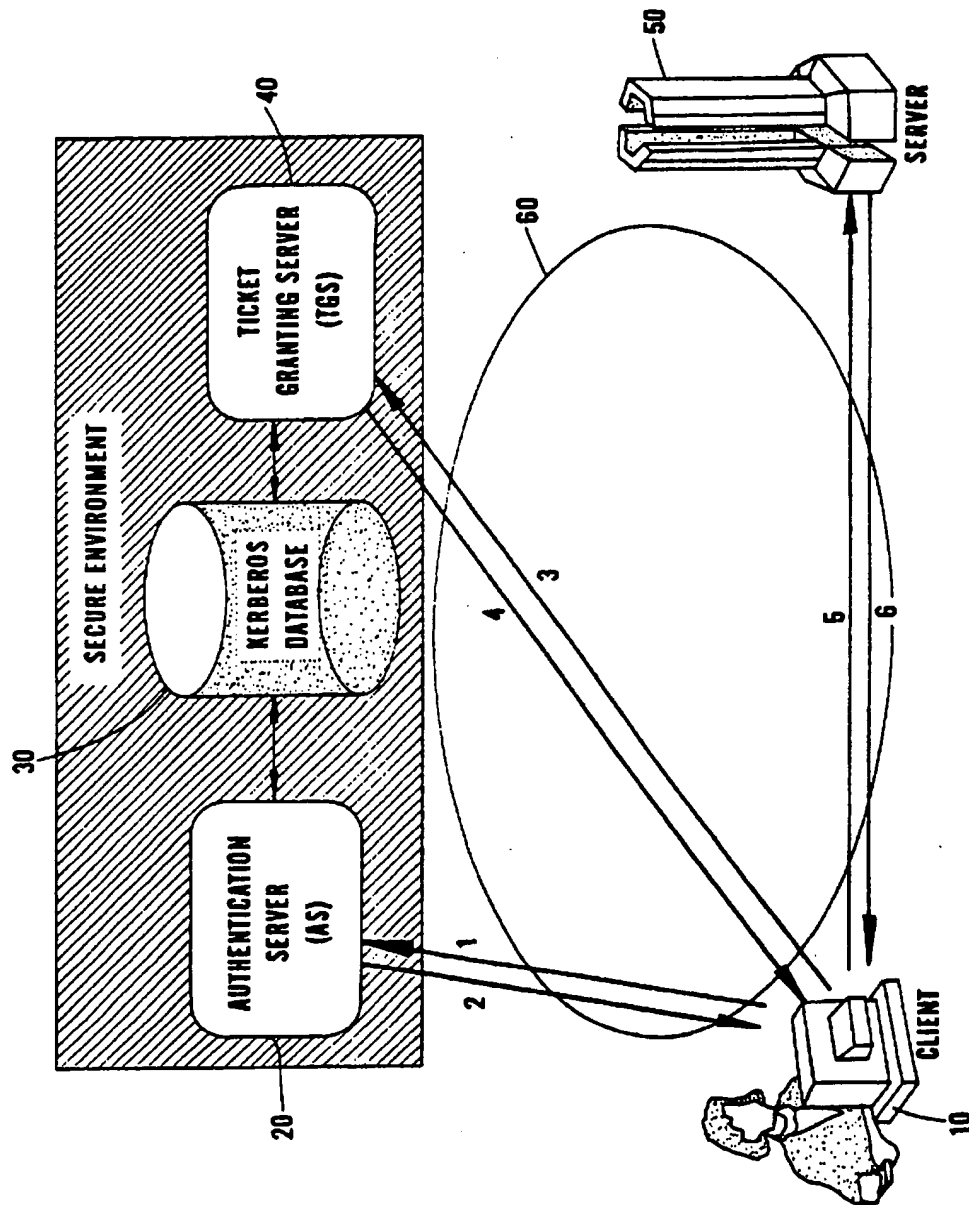


OTHER PUBLICATIONS

Schneier, B., "Applied Cryptography", Wiley & Sons, 1994, Sections 3.4 and 16.3 and p. 576. Multiple Key Public Key.  
R.L. Rivest, A. Shamir & L. Adleman, "A Method for

Obtaining Digital Signatures and Public-key Cryptosystems, CACM, vol. 21, pp. 120-126, Feb. 1978.

M.J. Wiener "Cryptoanalysis of Short RSA Secret Exponents", IEEE Transaction on Information Theory, vol. 36, No. 3, pp. 553-558.



**Figure 1**  
PRIOR ART

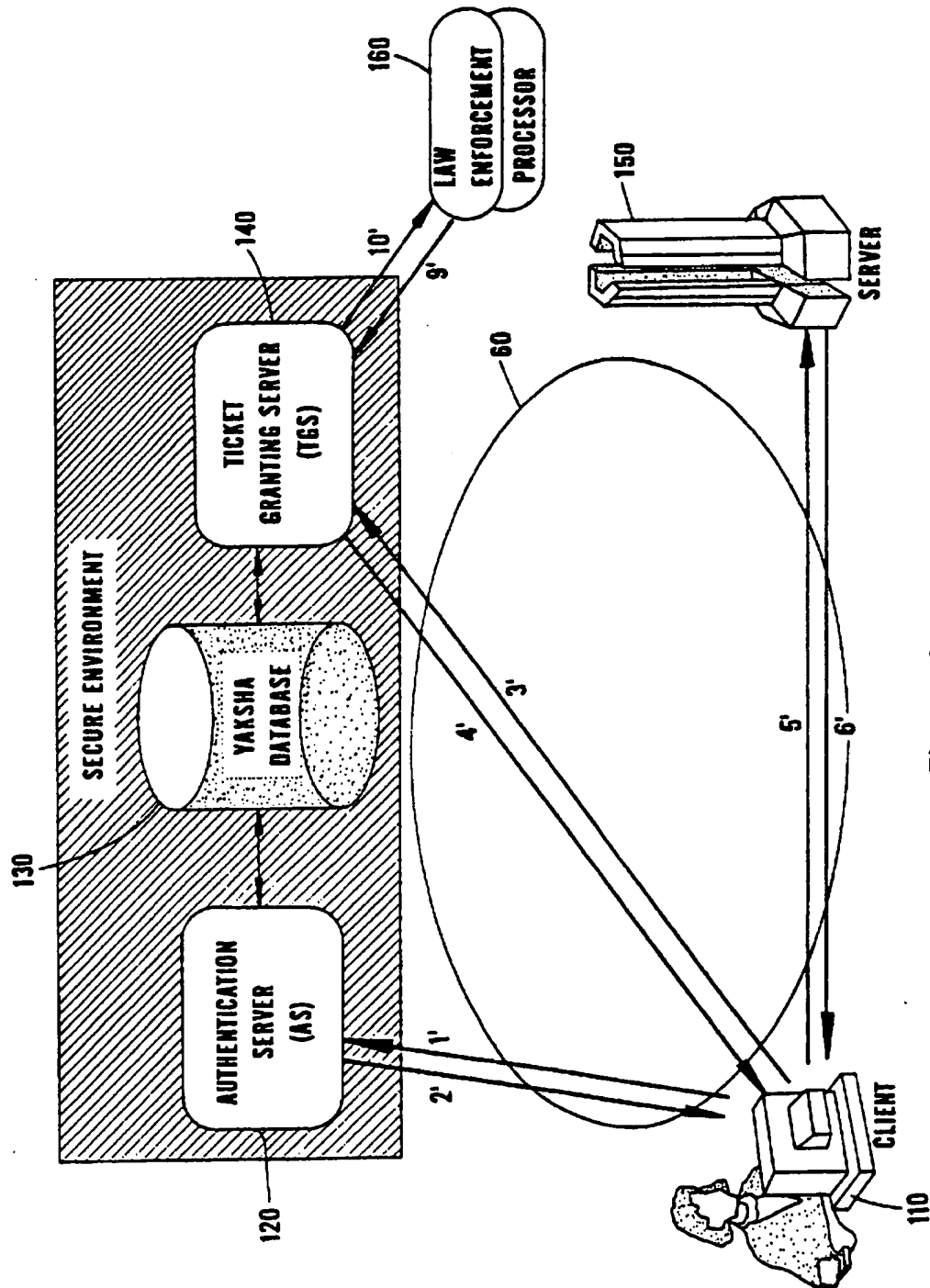


Figure 2

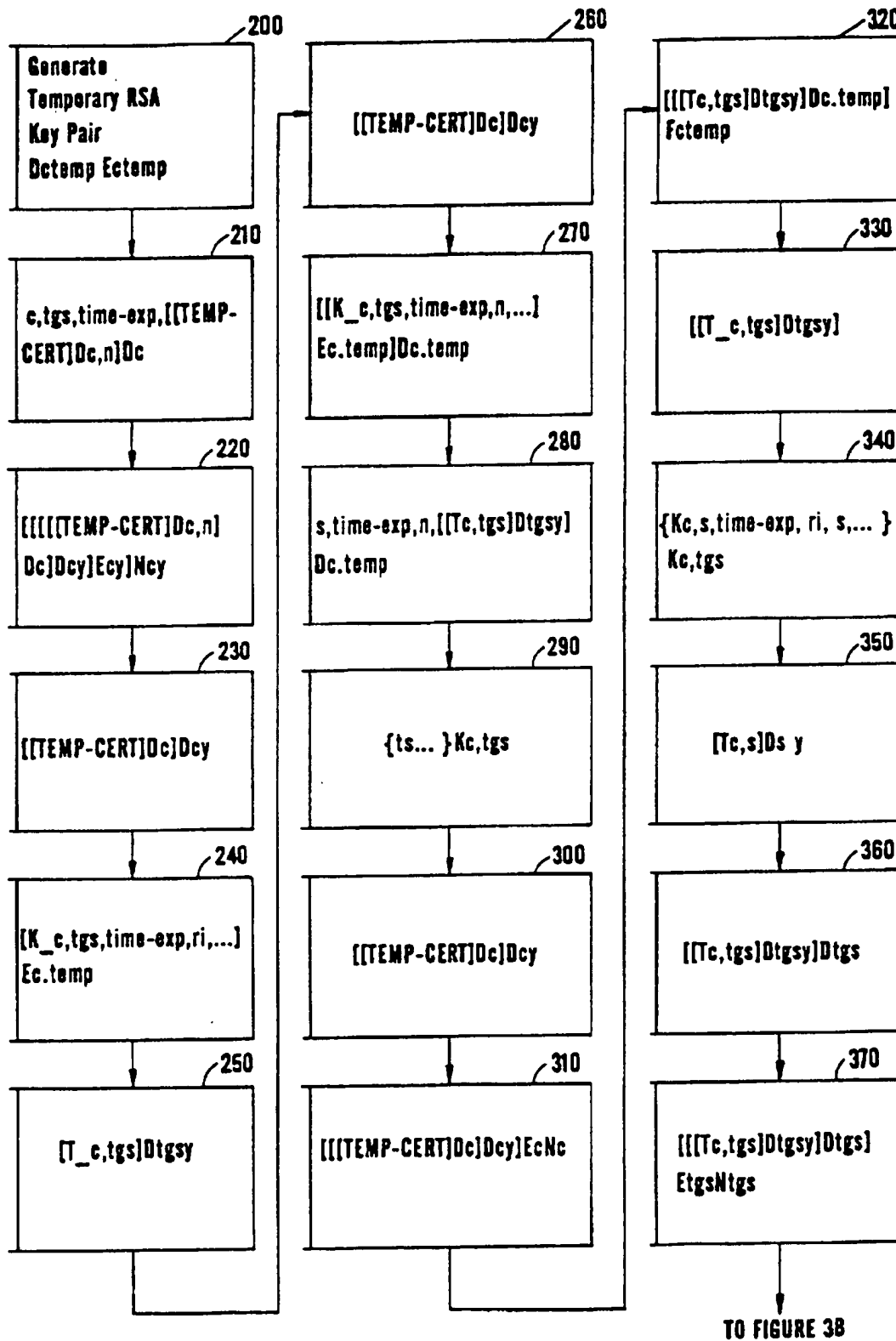
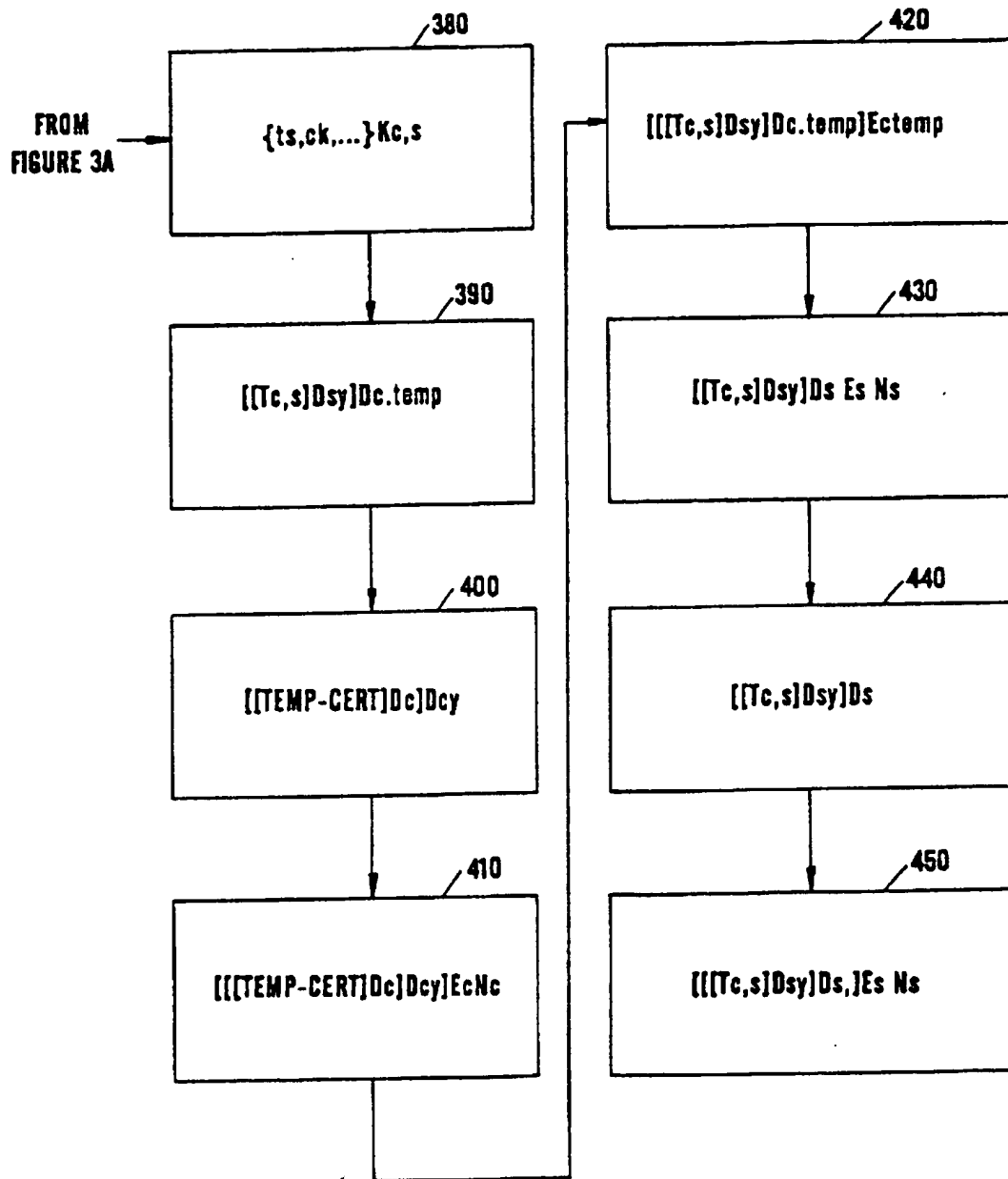
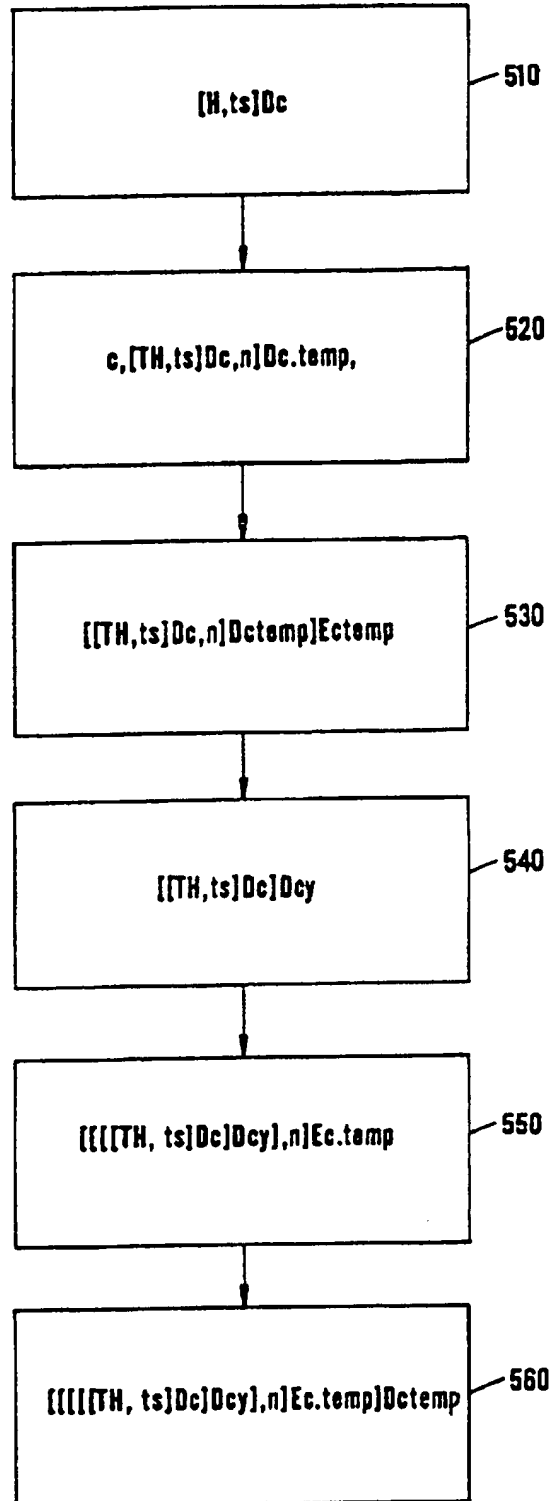
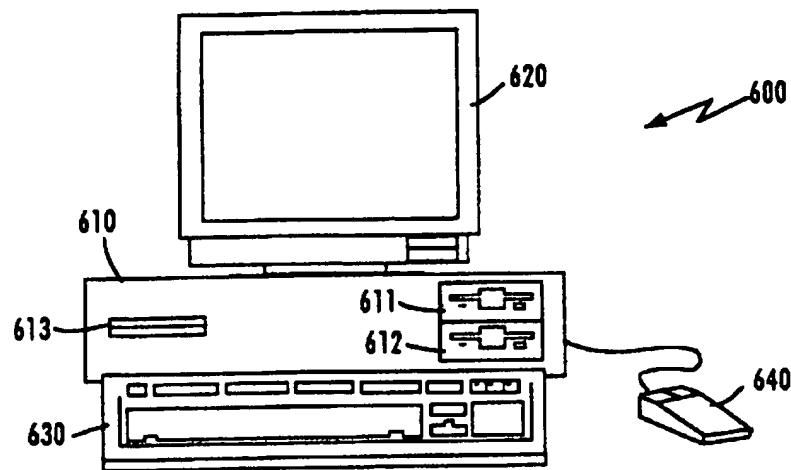
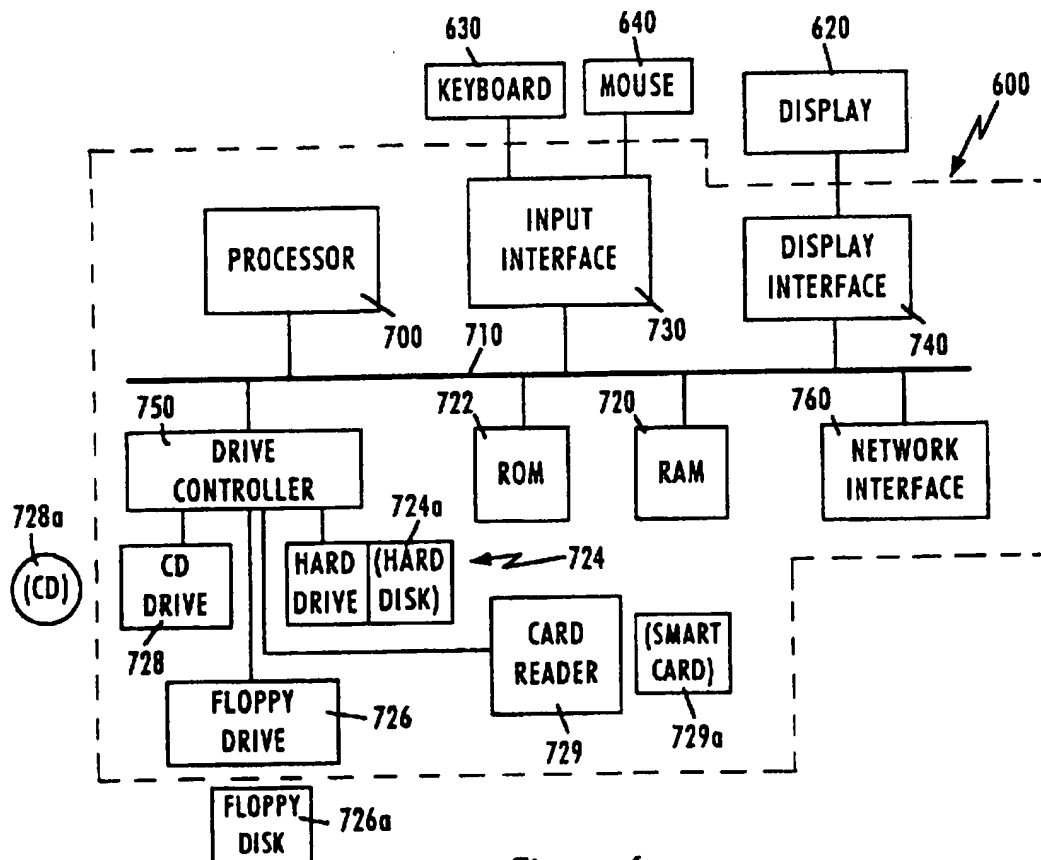


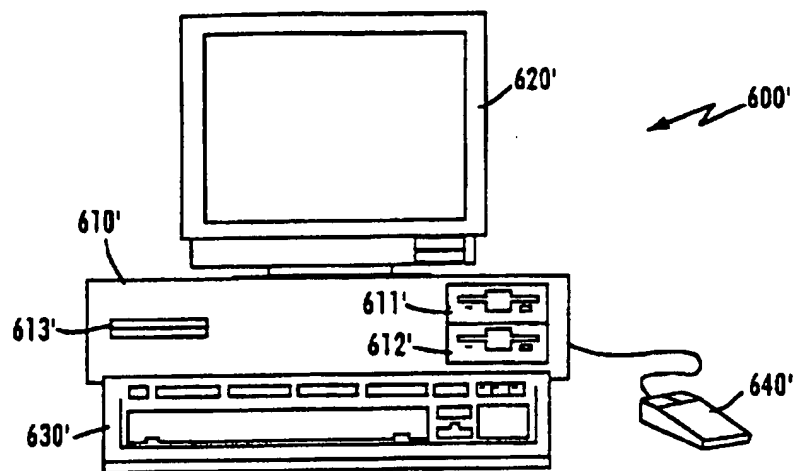
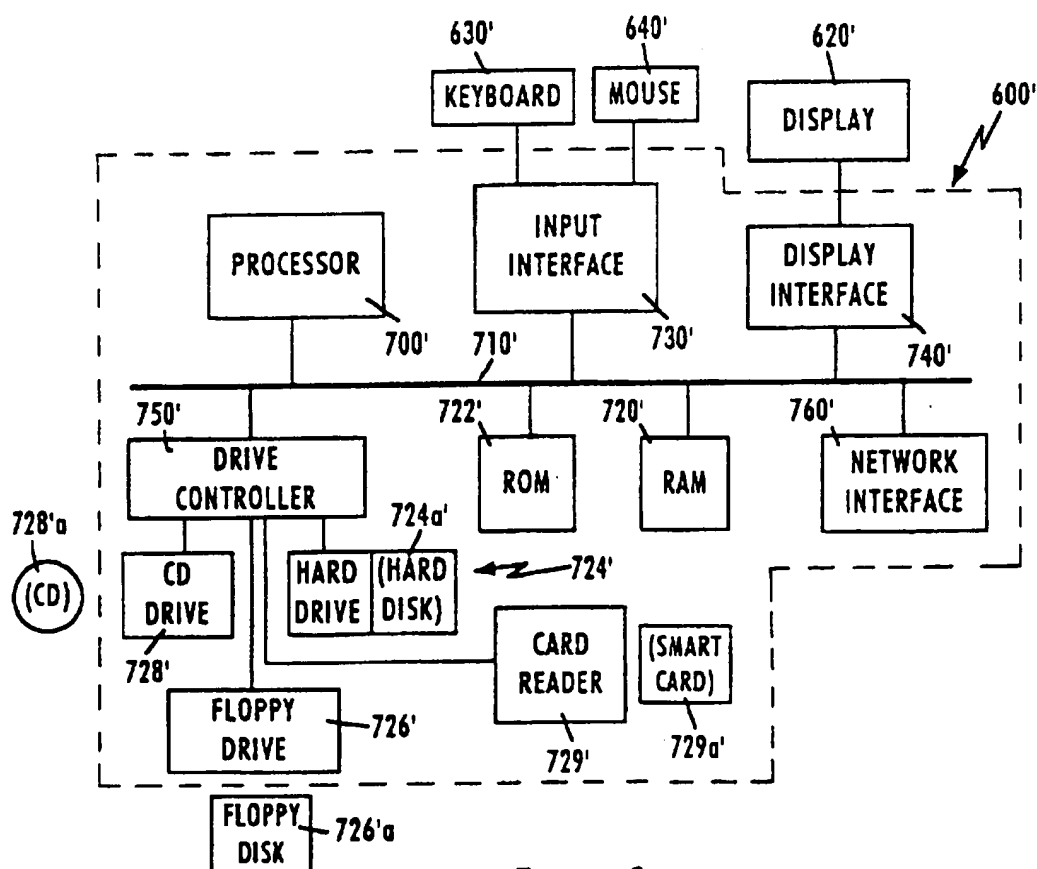
Figure 3A

*Figure 3B*

*Figure 4*

*Figure 5**Figure 6*



*Figure 7**Figure 8*

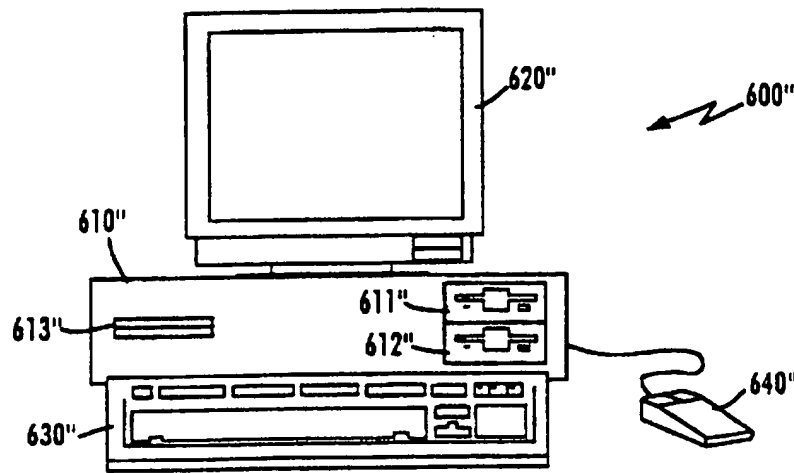


Figure 9

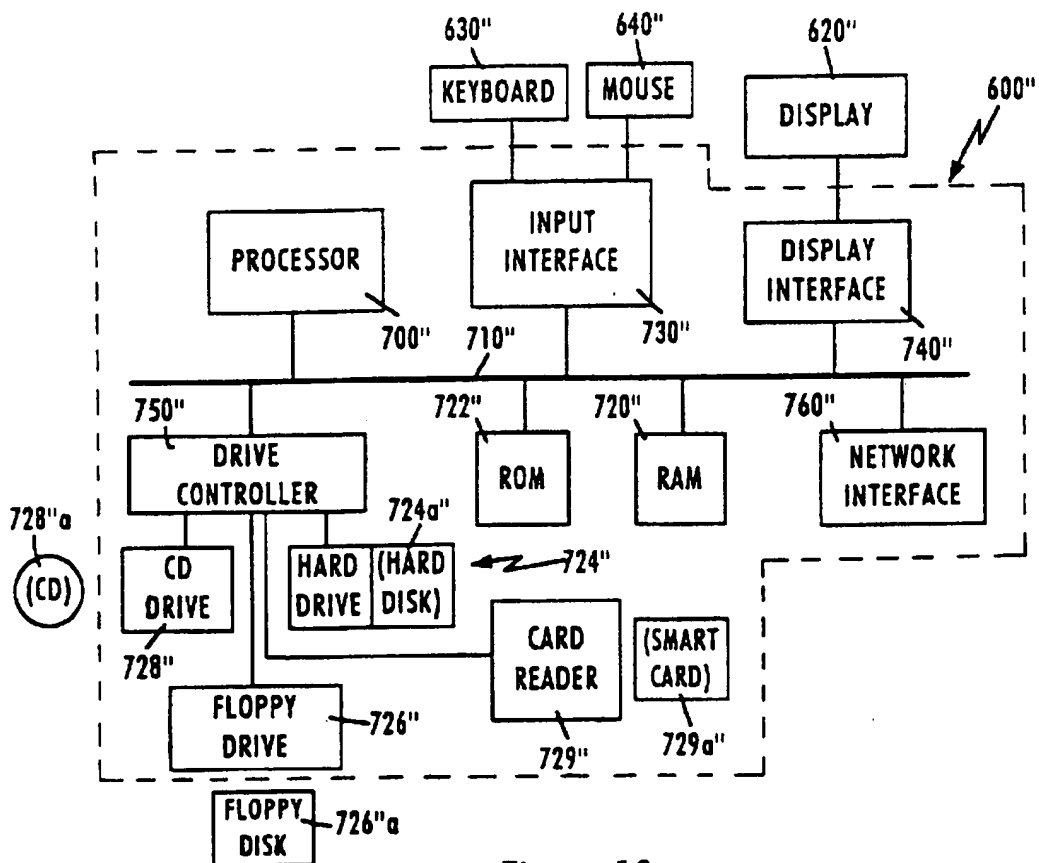
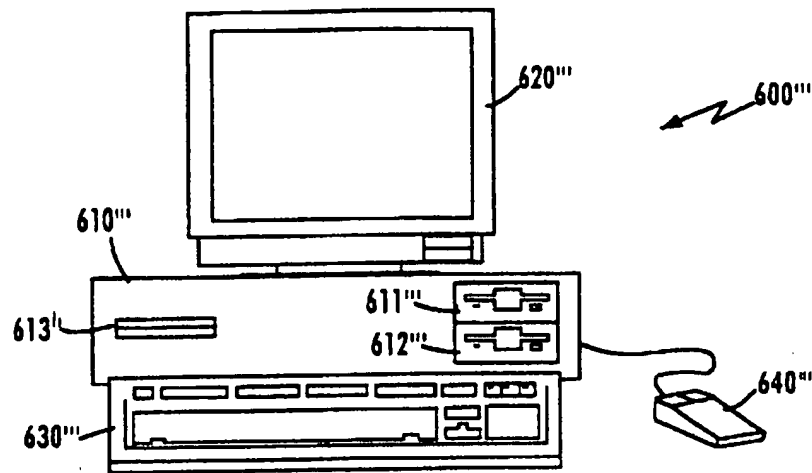
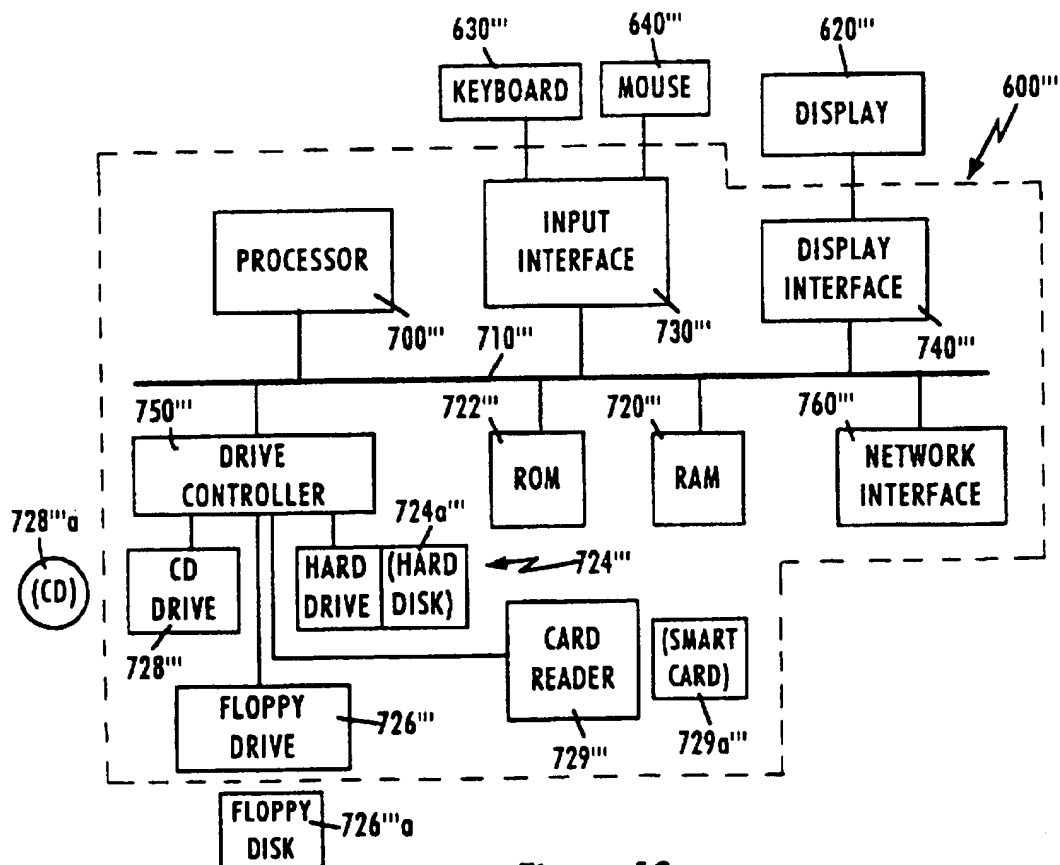


Figure 10

*Figure 11**Figure 12*

# COMPUTER SYSTEM FOR SECURING COMMUNICATIONS USING SPLIT PRIVATE KEY ASYMMETRIC CRYPTOGRAPHY

## RELATED APPLICATIONS

This application is a continuation-in-part of application Ser. No. 08/338,128, filed Nov. 9, 1994, now U.S. Pat. No. 5,535,276.

## BACKGROUND OF INVENTION

### 1. Field of the Invention

The present invention relates generally to securing communications using cryptography. More particularly, the present invention provides a computer and computer programming for enhancing the security of communications in an asymmetric crypto-system and is especially useful in enhancing communication security in conventional Kerberos authentication systems.

### 2. Description of the Related Art

Cryptosystems have been developed for maintaining the privacy of information transmitted across a communications channel. Often, a symmetric cryptosystem is used for this purpose. Symmetric cryptosystems, which utilize electronic keys, can be likened to a physical security system where a box has a single locking mechanism with a single key hole. One key holder uses his/her key to open the box, place a message in the box and relock the box. Only a second holder of the identical copy of the key can unlock the box and retrieve the message. The term symmetric reflects the fact that both users must have identical keys.

In more technical terms, a symmetric cryptosystem comprises an encryption function  $E$ , a decryption function  $D$ , and a shared secret-key,  $K$ . The key is a unique string of data bits to which the functions are applied. Two examples of encipherment/decipherment functions are the National Bureau of Standards Data Encryption Standard (DES) and the more recent Fast Encipherment Algorithm (FEAL). To transmit a message,  $M$ , in privacy, the sender computes  $C=E(M,K)$ , where  $C$  is referred to as the ciphertext. Upon receipt of  $C$ , the recipient computes  $M=D(C,K)$ , to recover the message  $M$ . An eavesdropper who copies  $C$ , but does not know  $K$ , will find it practically impossible to recover  $M$ . Typically, all details of the enciphering and deciphering functions,  $E$  and  $D$ , are well known, and the security of the system depends solely on maintaining the secrecy of key,  $K$ . Conventional symmetric cryptosystems are fairly efficient and can be used for encryption at fairly high data rates, especially if appropriate hardware implementations are used.

Asymmetric cryptosystems, often referred to as public key cryptosystems, provide another means of encrypting information. Such systems differ from symmetric systems in that, in terms of physical analogue, the box has one lock with two non-identical keys associated with it. For example, in an RSA system, either key can be used to unlock the box to retrieve a message which has been locked in the box by the other key. However, the system could be limited to using the keys in a particular sequence, such that the box can only be locked with the one key and unlocked with the other key.

In public key electronic cryptosystems, each entity, has a private key,  $d$ , which is known only to the entity, and a public key,  $E_n$ , which is publicly known. Once a message is encrypted with a user's public-key, it can only be decrypted using that user's private-key, and conversely, if a message is encrypted with a user's private-key, it can only be decrypted

using that user's public-key. It will be understood by those familiar with the art that although the terms "encrypt" and "decrypt" and derivations thereof are used herein in describing the use of public and private keys in an asymmetric public key cryptosystem, the term "transform" is commonly used in the art interchangeably with the term "encrypt" and the term "invert" is commonly used in the art interchangeably with the term "decrypt". Accordingly, as used herein in describing the use of public and private keys, the term "transform" could be substituted for the term "encrypt" and the term "invert" could be substituted for the term "decrypt".

If sender  $x$  wishes to send a message to receiver  $y$ , then  $x$ , "looks-up"  $y$ 's public key  $E_n$ , and computes  $M=E(C,e_y)$  and sends it to  $y$ . User  $y$  can recover  $M$  using its private-key  $d_y$ , by computing  $C=D(M, d_y)$ . An adversary who makes a copy of  $C$ , but does not have  $d_y$ , cannot recover  $M$ . However, public-key cryptosystems are inefficient for large messages.

Public-key cryptosystems are quite useful for digital signatures. The signer,  $x$ , computes  $S=E(M,d_x)$  and sends  $[M,S]$  to  $y$ . User  $y$  "looks-up"  $x$ 's public-key  $e_x$ , and then checks to see if  $M=D(S,e_x)$ . If it does, then  $y$  can be confident that  $x$  signed the message, since computing  $S$ , such that  $M=D(S,e_x)$ , requires knowledge of  $d_x$ ,  $x$ 's private key, which only  $x$  knows.

Public-key cryptography also provides a convenient way of performing session key exchange, after which the key that was exchanged can be used for encrypting messages during the course of a particular communications session and then destroyed, though this can vary depending on the application.

One public key cryptographic system is the Rivest, Shamir, Adleman (RSA) system, as described in Rivest, Shamir and Adleman, "A Method of Obtaining Digital Signatures and Public Key Cryptosystems", CACM, Vol 21, pp 120-126, February 1978. RSA is a public-key based cryptosystem that is believed to be very difficult to break. In the RSA system the pair  $(e,N_i)$ , is user  $i$ 's public-key and  $d_i$  is the user's private key. Here  $N_i=pq$ , where  $p$  and  $q$  are large primes. Here also  $e,d_i=1 \bmod \phi(N_i)$ , where  $\phi(N_i)=(p-1)(q-1)$  which is the Euler Totient function which returns the number of positive numbers less than  $N_i$  that are relatively prime to  $N_i$ . A Carmichael function is sometimes used in lieu of a Euler Totient function.

To encrypt a message being sent to user  $j$ , user  $i$  will compute  $C=M^{(e_i)} \bmod N_i$  and send  $C$  to user  $j$ . User  $j$  can then perform  $M=C^{(d_j)} \bmod N_j$  to recover  $M$ . User  $i$  could also send the message using his signature. The RSA based signature of user  $i$  on the message,  $M$ , is  $M^{(d_i)} \bmod N_i$ . The recipient of the message, user  $j$ , can perform  $(M^{(d_i)} \bmod N_i)^{(e_i)} \bmod N_i$ , to verify the signature of  $i$  on  $M$ .

In a typical mode of operation,  $i$  sends  $j$ ,  $M^{(d_i)} \bmod N_i$  along with  $M$  and a certificate  $C=(i,e_i,N_i)^{(d_{CA})} \bmod N_{CA}$ , where  $C$  is generated by a Certificate Authority (CA) which serves as a trusted off-line intermediary. User  $j$  can recover  $i$ 's public key from  $C$ , by performing  $C^{(e_{CA})} \bmod N_{CA}$ , as  $e_{CA}$  and  $N_{CA}$  are universally known. It should also be noted that in an RSA system the encryption and signatures can be combined.

Modifications to RSA systems have been proposed to enable multi-signatures to be implemented. Such an approach is described in "Digital Multisignature", C. Boyd, Proceedings of the Inst. of Math. and its Appl. on Cryptography and Coding, 15-17 December 1986. The proposed approach extends the RSA system by dividing or splitting the user private key  $d$  into two or more portions, say  $d_a$  and  $d_b$ , where  $d_a * d_b = d$ .

"A Secure Joint Signature and Key Exchange System", now U.S. Pat. No. 5,588,061, see also U.S. patent applica-

tion Ser. No. 08/277,808, which is also assigned to the assignee of the present application, modified Boyd's system, and made four significant additional points regarding split private key asymmetric cryptosystems. Although specifically applied to the two party case, the findings can be utilized more generally. The first point is that, assuming all operations are modulo  $N$ , breaking the joint signature system is equivalent to breaking RSA. This is true whether the attacker is an active or passive eavesdropper or one of the system users. It is assumed that key generation is conducted by a trusted third party, for example a tamper proof chip, and the factors of the RSA modulus  $N$  and  $\phi(N)$  are discarded after key generation and not known to any of the system users. The second point is the description of the following key exchange protocol: User 1 sends  $c_1 = m_1^{d_1}$  to User 2. User 2 recovers  $m_1 = c_1^{d_2}$ . Similarly User 2 transmits  $m_2$  to User 1. Each user then computes  $m = f(m_1, m_2)$ , where  $f$  is a function like XOR. Page and Plant prove mathematically that breaking this scheme is equivalent to breaking RSA. Again this is true whether the attacker is an active or passive eavesdropper or one of the system users. The third point is the introduction of the concept that one of the two users is a central server which maintains one portion of every user's RSA private key. In order to sign a message the user must interact with this server which, it is shown, cannot impersonate the user. Having to interact with such a central server has several important practical advantages, including instant revocation without difficult to maintain Certificate Revocation Lists (CRL). Kent, S., "Privacy Enhancement for Internet Electronic Mail: Part II: certificate Based Key Management", INTERNET RFC 1422, February 1993, a central point for audit and, as discussed below, a method of providing for digital signatures in an era where smart cards are not yet ubiquitous. Finally, the paper also proves mathematically that even if one of the two portions,  $d_1$ , and  $d_2$ , of the private key,  $d$  is short, say 64 bits, an eavesdropper will have equal difficulty breaking the split key system as would be experienced in breaking RSA. As a consequence, a digital signature infrastructure can be built where users who remember short (8-9 characters) passwords, can interact with the central server to create RSA signatures which are indistinguishable from those created using a full size private key stored on a smart card.

One symmetric cryptosystem is the Kerberos authentication system, Kohl, J. T. and B. C. Neuman, "The Kerberos Network Authentication Service", INTERNET RFC 1510, September 1993, which is based on the classic Needham-Schroeder authentication protocols, Needham, R. M. and Schroeder M. D., "Using Encryption for Authentication in Large Networks of Computers", Communications of the ACM, v. 21, n. 12, December 1978, with extensions by Denning-Sacco, D. E. Denning and G. M. Sacco, "Timestamps in Key Distribution Protocols," Communications of the ACM, v. 24, n. 8, August 1981, pp. 553-536. The system uses a trusted third party model to perform authentication and key exchange between entities in a networked environment, for example, over a local or wide area network. Kerberos uses symmetric key cryptosystems as a primitive, and initial implementations use the Data Encryption Standard (DES) as an interoperability standard, though any other symmetric encryption standard can be used. After close to a decade of effort, the Kerberos authentication system is now a fairly mature system whose security properties have held up fairly well to intense scrutiny. Further, vendors are now delivering Kerberos as a supported product. Kerberos has also been adopted as the basis for the security service by the Open Software Foundation's (OSF) Distrib-

uted Computing Environment (DCE). Consequently, Kerberos can be expected to be among the most widespread security systems used in distributed environments over the next several years.

For the sake of clarity, a "simplified" version of the Kerberos protocol described by Neuman and Ts'o in Neuman, B. C. and Ts'o, T., "Kerberos: An Authentication Service for Computer Networks", IEEE Communications, September 1994, will be discussed below. The complete protocol is described in Kohl, J. T. and Neuman, B. C., "The Kerberos Network Authentication Service", INTERNET RFC 1510, September 1993. Further, the following discussion is based on Neuman, B. C. and Ts'o, T., "Kerberos: An Authentication Service for Computer Networks", IEEE Communications, September 1994, and for the sake of consistency uses almost the same notation. The fundamental message exchanges are shown in FIG. 1. In message 1 the user uses a personal computer or workstation 10 to request a ticket granting ticket (TGT) from an authentication server (AS) 20. The server 20 creates such a ticket TGT, looks up the user's password from the Kerberos database 30, encrypts the TGT with the password and sends it to the user via the computer 10 in message 2. The user decrypts the TGT with her password using computer 10, and stores the TGT on computer 10, for example on a hard disk or in the random access memory (RAM). Then, when the user desires to access a service, she sends message 3, which contains the TGT to the ticket granting server 40. The server 40 verifies the TGT and sends back, in message 4, a service ticket to access the service server 50, and a session key, encrypted with the user's password retrieved from database 30. In message 5 the user presents via computer 10 the service ticket to the server 50, which verifies it and also recovers the session key from it. If mutual authentication is required, the server 50, in message 6, sends back a message encrypted with the session key. All communications between servers 20, 40 and 50 and computer 10 are via network 60. All communications between servers 20 and 40 and database 30 are preferably by direct communications link.

The Kerberos messages will now be described in further detail. Message 1 known as `_req` (request to authentication service), consists of:

`as_req: c, tgs, time-exp, n` (1)

where  $c$  is the name of the client (user), and  $tgs$  is the name of the ticket granting service associated with server 50, for which the client is requesting a ticket granting ticket and  $time-exp$  is the requested expiry time of the ticket, e.g. eight hours, and  $n$  is a fresh random number. This message is sent from computer 10 in the clear, and all parts of it are visible to an eavesdropper. The authentication server 20 responds with Message 2, with

`as_rep: {Kc, tgs, time-exp, n, . . . } Kc, {Tc, tgs} Ktgs` (2)

where  $Kc, tgs$  is the symmetric session key to be shared between the ticket granting server ( $tgs$ ) 40 and the user for the lifetime of this ticket.  $Kc, tgs$  and the other information is encrypted with symmetric key  $Kc$  which is the user's password, i.e. the long term secret which is shared with the Kerberos server. Only a user who knows  $Kc$  will be able to decrypt this message to obtain  $Kc, tgs$ . The key  $Kc, tgs$  is also embedded in the ticket  $Tc, tgs$ , which in the `as_rep` is encrypted using  $Ktgs$ , a long term key known only to the server 20 and the server 40. After decrypting the first part of the message on computer 10, the user stores the data received in the `as_rep` on computer 10. The main purpose of

5

this process is to avoid storing the long term key Kc on the computer 10 where it may be compromised. Rather, the key Kc.tgs is used in subsequent communications in lieu of Kc. Since Kc.tgs is relatively short lived, the damage an attacker can cause by learning this key is significantly less than the damage which might be caused by compromise of long term key Kc. It is worth observing that the server 20 does not verify the identity of the user before responding to a user's as\_req with a as\_rep. Rather server 20 relies on the fact that to be able to make any use of the as\_rep, the recipient must know Kc. So not only can an attacker eavesdrop on the network to recover as\_rep, but can actually get an as\_rep from the server 20 by sending a fraudulent as\_req. The attacker can then take the portion of the as\_rep encrypted with Kc, and attempt to decrypt by taking guesses at Kc. Since Kc is typically a user selected password, Kc may well be a poor password, which the attacker can guess.

When the client wishes to obtain a ticket to access server 50, it sends to the server 40, Message 3,

tg\_req:s,time-exp,n,{Tc,tgs}Ktgs,{ts...}Kc,tgs (3)

This message consists of the name of the server 50, s, the requested expiry time, time-exp, and the random number n, in clear text. It also contains the encrypted ticket granting ticket {Tc,tgs}Ktgs which was received by the client computer 10 in the as\_rep message. The server 40, which knows Ktgs, can decrypt and recover Tc,tgs, which is a valid ticket. In order to prevent a replay attack in which an attacker might gain some benefit by resending a valid {Tc,tgs}Ktgs at a later time, the tg\_req message also contains an authenticator, which is a time stamp, ts, a check sum and other data, all encrypted with the session key Kc.tgs. Since this session key is embedded in the ticket Tc,tgs, which the server 40 has recovered, the server 40 can decrypt the authenticator and verify the time stamp and check sum, etc. By maintaining a cache of recently received authenticators, the server 40 can detect replays.

Having verified the authenticity of the tg\_req, the server 40 responds with Message 4,

tg\_rep:{Kc,s,time-exp,n,s,...}Kc,s{Ts,s}Ks (4)

This message is very similar in structure and purpose to the as\_rep, message. The first part consists of a session key, expiry time, etc., encrypted with Kc.tgs. The client computer 10 can decrypt this to recover the session key and other information. The second portion is a ticket to access the server 50, encrypted with the long term key Ks shared by the server 50 and the server 40. The client using computer 10 now constructs Message 5 and sends it to the server 50, as follows:

ap\_req:{ts,ck,...}Kc,s{Ts,s}Ks (5)

This message is similar to the tg\_req, in that it contains an encrypted ticket {Ts,s}Ks which the server 50 can use to recover Ts,s, which authenticates the client to the server 50 and, among other information, contains the session key Kc,s. The server 50 then uses Kc,s to decrypt the first part of the message, the authenticator, which has a time-stamp, ts, a check-sum, ck, etc.

Having verified the authenticity of the client, the client computer 10 and server 50 are ready to communicate. However, in some cases the client may request mutual authentication, in which case the server 50 must first respond with message 6,

ap\_rep:{ts}Kc,s (6)

6

which is basically proof that the server 50 successfully recovered Kc,s from the ticket Tc,s, which means the server knew Ks, which in turn is proof of authenticity of the server. The actual protocol has a number of options and is more complex, but the basic structure is defined by these six messages. Those interested are referred to Kohl, J. T. and B. C. Neuman, "The Kerberos Network Authentication Service", INTERNET RFC 1510, September 1993, for more details.

Kerberos does have limitations, and among the more serious ones are (i) compromise of the central trusted on-line Kerberos server, or the central Kerberos database, is catastrophic, since it retains long term user secrets, (ii) Kerberos is vulnerable to password guessing dictionary attacks, and (iii) Kerberos does not provide non-repudiation services, i.e. digital signatures. The first limitation is intrinsic to the Needham Schroeder protocol when used with symmetric cryptosystems like DES. The second problem is significant because experience suggests that password guessing attacks tend to be far more common than most other forms of attacks, since they are simple and effective. Finally, Kerberos was designed to provide authentication and key-exchange, but it was not designed to provide digital signatures. However, organizations using Kerberos may also need to implement digital signatures, and must now maintain separate security infrastructures for conventional Kerberos and for digital signatures, which accordingly results in significant additional costs.

Digital Equipment Corporation's SPX system, Tardo, J., and K. Alagappan, "SPX Global Authentication Using Public-Key Certificates", Proceedings of the 1991 IEEE Symposium on Research in Security and Privacy, 1991, is an example of a system with a public key infrastructure which achieves many of the same goals as Kerberos without its associated limitations. However, the SPX system does not maintain the standard Kerberos authentication system whose security properties have been widely examined. Therefore the SPX system is substantially different than the Kerberos protocol and the Kerberos source tree. In particular, the SPX system's protocol is sufficiently different from Kerberos to make integration of these systems require a complete reworking of the Kerberos protocol.

Bellovin and Merritt's Encrypted Key Exchange (EKE), Bellovin, S. M. and M. Merritt, "Encrypted Key Exchange: Password-Based Protocols Secure Against Dictionary Attacks", Proceedings of the 1992 IEEE Computer Society Conference on Research in Security and Privacy, 1992, can potentially be integrated with Kerberos to prevent dictionary attacks. However, the EKE multi-pass protocol would require very significant changes to the Kerberos system. EKE assumes that the participants share a common long term secret.

It has been suggested, by at least one expert, Kohl, J. T., "The Evolution of the Kerberos Authentication Service", BurOpen Conference Proceedings, May 1991, as quoted in Schneier, B., Applied Cryptography: Protocols, Algorithms and Source Code in C, John Wiley and Sons, New York, 1994) that: "Taking advantage of public-key cryptography would require a complete reworking of the [Kerberos] protocol".

It will perhaps also be worthwhile to describe the taxonomy of dictionary type attacks on system security. Dictionary attacks are a common form of attack, and it is well-known that many systems (e.g. UNIX or Kerberos), Morris, R. and K. Thompson, "Password Security: A Case History", Communications of the ACM, 22(11), November 1979, are vulnerable, Karn, P. R. and D. C. Feldmeier, "UNIX password security—Ten years later", Advance in

Cryptology—CRYPTO 89. G. Brassard (Ed.) Lecture Notes in Computer Science, Springer-Verlag, 1990, to them. However, all dictionary attacks are not alike.

There are four parameters to a dictionary attack: The first is the known plain text,  $S$ , which can take two forms. The first form is a string  $S1$  which is known in advance to the attacker. An example of  $S1$  is a string of zeroes. The second form is a string  $S2$  which is not known to the attacker in advance, but which will be known when the attack is successful. An example of  $S2$  is any string with some form of predictable redundancy, for instance, a time stamp. Another example is a number with particular, easily tested, mathematical properties, for instance, a prime, or a non-prime with no small factors. The second parameter is the ciphertext  $C$ , typically of the form  $F(S,k)$  where  $k$  is the password being sought. The third parameter is the password space  $P$  being guessed at. The attacker will take guesses  $p_1, p_2, \dots, p_n$ , until a  $p_i$  which is equal to  $k$  is found. The fourth parameter is the function  $F$  and its inverse, assuming one exists, which are typically public information. Those skilled in the art will recognize that important distinctions exist between cases when  $F$  is an RSA or similar function rather than a DES or similar function.

These four parameters result in at least two distinct forms of dictionary attacks. The first is  $S1$  type attacks. Here the attacker typically computes  $F(S1, p_i)$  on all passwords in  $P$  until a  $p_i$  where  $F(S1, p_i) = C$  is uncovered. This is the most dangerous form of attack since the attacker can (i) recompute the  $F(S1, p_i)$  for all or many  $p_i$  and (ii) amortize his attack against several users. UNIX is particularly vulnerable to such attacks. The second form of attack is  $S2$  type attacks. Here the attacker is typically computing  $F^{-1}(C, p_i)$  and hoping to find an  $S2$  which can be recognized. The attacker cannot start computations before  $C$  is captured. Further, since  $C$  will be different for each instance, no amortizations of the attack are possible. The Kerberos system is vulnerable to this form of attack.

#### OBJECTIVES OF THE INVENTION

A need exists for a programmed computer and computer programming instructions which can secure communications such that the compromise of a central database, such as the database in a conventional Kerberos system, will not be catastrophic to overall system security, that is the attacker will not be able to use a compromised password or crypto-key to impersonate a user.

A need also remains for a programmed computer and computer programming instructions which can secure communications such that the communications are not vulnerable to dictionary attacks.

A still further need exist for a programmed computer and computer programming instructions which can secure communications such that one user is authenticated to another user.

Yet another need exists for a programmed computer and computer programming instructions which can secure communications such that digital signatures are facilitated, to provide for non-repudiation.

Additionally needed is a programmed computer and computer programming instructions which can secure communications such that security of conventional Kerberos systems is enhanced with minimum changes to the standard Kerberos protocol.

Another need which continues to exist is for a programmed computer and computer programming instructions which can secure communications such that the use of "smart cards" is facilitated.

A programmed computer and computer programming instructions are needed which allows the reuse of an authentication infrastructure for digital signature, that is the same key(s) should be available for both authentication and digital signatures and only a single secure database should be required for key storage.

Additional needs which can be satisfied by, as well as other advantages and novel features of, the present invention will become apparent to those skilled in the art from this disclosure, including the following detail description, as well as by practice of the invention. While the invention is described below with reference to preferred embodiments, it should be understood that the invention is not limited thereto. Those of ordinary skill in the art having access to the teachings herein will recognize additional applications, modifications and embodiments in other fields, which are within the scope of the invention as disclosed and claimed herein and with respect to which the invention could be of significant utility.

#### SUMMARY OF THE INVENTION

According to the present invention, programmed computer(s), such as a personal computer(s), workstation(s) or other processing device(s), and computer programming are provided for securing communications over a system having a plurality of users. Each system user has an associated asymmetric crypto-key, such as an RSA crypto-key, with a public key portion and a corresponding private key portion. Each public key portion is available to the plurality of system users, and each private key portion has a first private key portion, which is preferably short in length, e.g. 8 to 12 characters, known only to the associated user and a corresponding second private key portion.

To enhance the security of communications between users of a crypto-system, computer programming stored on a tangible medium is read by a first user's computer and thereby causes the computer to perform in the following manner. In accordance with its stored programming instructions a first user's computer generates a temporary asymmetric crypto-key, having a temporary private key portion and an associated temporary public key portion. The temporary public key portion is encrypted, by the first user's computer, with the first private key portion of a crypto-key associated with the first user to form a first encrypted message.

Computer programming stored on a tangible medium is read by a second user's computer, e.g. a security or authentication server, and thereby causes the computer to perform in the following manner. The second user's computer, in accordance with its stored programming instructions, obtains the temporary public key portion by applying the second private key portion, typically retrieved from a secured database, which may be stored on a storage device within the second user's computer, and the public key portion of the first user crypto-key to the first encrypted message. This authenticates the first user to a second user. The second user's computer now further encrypts the first encrypted message with the second private key portion of the first user crypto-key to form a second encrypted message, which serves as a certificate having the temporary public key signed with both permanent private key portions of the first user's private key.

The first user's computer is now driven by the programming stored thereon to apply the public key portion of the first user crypto-key to decrypt the second encrypted message so that the first user's computer obtains the temporary public key portion, thereby authenticating the second user to the first user.

The second user's computer is next driven by the programming stored thereon to encrypt a message to a third user, such as an instruction to provide a ticket for service or the desired service itself, and a first symmetric session crypto-key, which might also be generated by the second user's computer in accordance with the programming instructions, with the second private key portion of a third user crypto-key associated with the third user to form third encrypted message. Like the second private key portion of the first user crypto-key, the second private key portion of the third user crypto-key is preferably accessible and retrieved from a secured database. The third user could, for example be a ticket granting server in a Kerberos system or a service server of any type which might require secure communications.

Computer programming stored on a tangible medium is read by a third user's computer, e.g. a ticket granting or service server, and thereby causes the computer to perform in the following manner. The third user's computer is driven by the programming stored thereon to encrypt the first symmetric session crypto-key with the temporary public key portion to form a fourth encrypted message.

The first user's computer is then driven by the programming stored thereon to obtain the first symmetric session crypto-key by applying the temporary private key portion, which only should only be available to the first user's computer, to decrypt the fourth encrypted message. The first user's computer is also operated to further encrypt the third encrypted message with the temporary private key portion to form a fifth encrypted message. Additionally, the first user's computer encrypts a second message, such as a time stamp, with the first symmetric session key to form a sixth encrypted message.

The third user's computer is next driven by the programming stored thereon to obtain the first message and the first symmetric session crypto-key by applying the temporary public key portion and the first private key portion of the third user crypto-key to decrypt the fifth encrypted message and thereby authenticate the first user and the second user. The third user's computer is also instructed by the programming to obtain the second message by applying the first symmetric session crypto-key to the sixth encrypted message.

In accordance with the stored programming instructions, the third user's computer now encrypts a third message to the fourth user, for example a service server, and a second symmetric session key with the second private key portion of a crypto-key associated with the fourth user to form a seventh encrypted message. The second private key portion of the fourth user crypto-key is typically also retrieved from a secured database. The third user's computer is also driven by the programming to encrypt the third encrypted message with the first private key portion of the third user crypto-key to form an eight encrypted message. Still further, the third user's computer is operated to additionally encrypt the second symmetric session key with the first symmetric session key to form a ninth encrypted message.

The first user's computer is driven by the programming stored thereon to obtain the first message and the first symmetric session key by applying the public key portion of the third user crypto-key to the eight encrypted message. This authenticates the third user to the first user. The first user's computer is also driven to obtain the second symmetric session key by applying the first symmetric session key to the ninth encrypted message. The seventh encrypted message is further encrypted by the first user's computer

with the temporary private key portion to form a tenth encrypted message. Additionally, the first user's computer is operated to encrypt a fourth message to the fourth user with the second symmetric session key to form a eleventh encrypted message.

Computer programming stored on a tangible medium is read by a fourth user's computer, e.g. a service server, and thereby causes the computer to perform in the following manner. In accordance with the stored programming instructions, the fourth user's computer obtains the third message and the second symmetric session crypto-key by applying the temporary public key portion along with the first private key portion of the fourth user crypto-key to the tenth encrypted message. This authenticates the first and third users to the fourth user. The fourth user's computer is next driven to apply the second symmetric session crypto-key to the eleventh encrypted message and thereby obtain the fourth message. The fourth user's computer is then operated to encrypt the seventh encrypted message with the first private key portion of the fourth user crypto-key to form a twelfth encrypted message.

The first user's computer is driven by the programming stored thereon to obtain the third message and the second symmetric session key by applying the public key portion of the fourth user crypto-key to the twelfth encrypted message. This authenticates the fourth user to the first user. The first user's computer is also instructed by its programming to apply the second symmetric session crypto-key to encrypt and decrypt data communicated between the first and fourth users.

It should be recognized that the third user's computer could, if desired, also be directed by its programming to provide the second session key to another user, for example the FBI, to allow legal eavesdropping on communications between the first and fourth users without revealing the permanent private key of the first and fourth user.

In accordance with yet other aspects of the invention, joint signatures can be performed by two users, for instance the first and second users described above, using the temporary crypto-key. The first user's computer is driven by the programming stored thereon to generate a hash message and a time stamp encrypted with the first private key portion of the first user's crypto-key to form an encrypted message. A signature of the first user is thereby placed on the hash message. The first user's computer is also driven to encrypt the encrypted hash and time stamp message with the temporary private key portion and with the first private key portion of the first user crypto-key to form a further encrypted message. The first user's computer is further driven by the stored programming to encrypt the temporary public key portion with the first private key portion of the first user crypto-key to form another encrypted message.

The second user's computer is driven by the programming stored thereon to decrypt and obtain the temporary public key portion by applying the second private key portion and the public key portion of the first user crypto-key. Pursuant to instructions from the stored programming, the second user's computer is next driven to apply the temporary public key portion, thus obtained, along with the second private key portion and the public key portion of the first user crypto-key to obtain the encrypted hash message and time stamp, that is the second user's computer partially rather than fully decrypts the message so that what remains is the hash message and time stamp encrypted with only the first private key portion of the first user's crypto-key, which as discussed previously is the signature of the first user on the hash message and time stamp.



As discussed above, the second user's computer is typically driven to retrieve the second private key portion of the first user's crypto-key from the secured database, such as that provided in conventional Kerberos systems. It also will be understood that the second user's computer will, in appropriate cases, also be driven to fully decrypt the message to ensure that the second user is agreeable to co-signing the message. The second user's computer, in accordance with its programming instructions, then further encrypts the partially decrypted message with the second private portion of the first user's crypto-key. The second user has thereby also signed the hash message and time stamp. The second user's computer is now operated to encrypt the jointly signed message with the second temporary key portion.

The first user's computer is driven by the programming stored thereon to partially decrypt the message by applying the temporary private key portion, thereby leaving only the jointly signed message, i.e. the hash message and time stamp encrypted with the first and second private key portions of the first user's crypto-key. A still further user's computer, in accordance with its programming, can now decrypt the hash message and time stamp using only the public key portion of the first user's crypto-key and thereby verify that the hash message and time stamp have been jointly signed.

According to still further aspects of the invention, the application of the public key portion of the first user crypto-key to decrypt the second encrypted message and obtain the temporary public key portion is performed multiple times so as to also authenticate the first and second users to the third user as well as the fourth user. Preferably, in order to defend against dictionary attacks, the first encrypted message includes a first random number string concatenated with the temporary public key portion, the fourth encrypted message includes a second random number string different from the first random number string concatenated with the first symmetric session key, and the ninth encrypted message includes a third random number string, different from the first and second random number strings, concatenated with the second symmetric session crypto-key. Likewise, the hash message and time stamp signed by the first user, i.e. encrypted with the first private key portion of the first user crypto-key preferably also has a random number string concatenated to it prior to being further encrypted with the first private key portion of the first user crypto-key and the temporary private key portion. Similarly, the jointly signed hash message and time stamp preferably has a different random number string concatenated to it prior to being further encrypted with the temporary public key portion. The asymmetric crypto-keys are applied using modular exponentiation and the temporary crypto-key has an associated expiration time.

It will be understood that various features and aspects of the present invention can be implemented together or separately as may be desired for the particular application. Hence, in accordance with the present invention, a computer or computer programming might, in appropriate circumstances, only perform authentication of a user and a third party, such as an authentication server. The user's computer, in such a case, encrypts a message with the first private key portion of the user's asymmetric crypto-key to form a first encrypted message. The third party's computer obtains the first message by applying the second private key portion, and typically the public key portion of the user's crypto-key, to the first encrypted message, and thereby authenticates the user to the third party. The third party's computer then encrypts the first encrypted message with the second private key portion of the user's crypto-key to form

a second encrypted message. The first user's computer obtains the first message by applying the public key portion of the user's crypto-key to decrypt the second encrypted message and thereby authenticates the third party.

Similarly, it may in certain cases be desirable only to require authentication of one user to another. This can be accomplished by one user's computer encrypting a first message with the first private key portion of the user's asymmetric user crypto-key to form a first encrypted message. The third party's computer encrypts the first encrypted message with the second private key portion of the user's crypto-key to form a second encrypted message. A second user's computer can now obtain the first message by applying the public key portion of the other user's crypto-key to decrypt the second encrypted message. This authenticates the first user to the second user and also verifies that the first message is signed by both the first user and the third party.

Authentication of one user to another can also be accomplished by the third party's computer encrypting a first message with the second private key portion of the first user's asymmetric crypto-key to form a first encrypted message. The first user's computer can now encrypt the first encrypted message with the first private key portion of his/her user crypto-key to form a second encrypted message. The second user's computer can obtain the first message by applying the public key portion of the first user's crypto-key to decrypt the second encrypted message and thereby authenticate the first user and verify that the first message is signed by both the first user and the third party.

In still other cases it may be desirable to limit the utilization of the present invention to temporary key distribution. In such a case, a user's computer first generates a temporary asymmetric crypto-key having a temporary private key portion and an associated temporary public key portion. The user's computer encrypts the temporary public key portion with the first private key portion of the user's crypto-key to form a first encrypted message. The third party's computer encrypts the first encrypted message with the second private key portion of the user's crypto-key to form a second encrypted message. Another user's computer can now obtain the temporary public key portion by applying the public key portion of the first user's long term crypto-key. Each user's computer can now encrypt and decrypt communications between the two users with one of the temporary key portions.

In yet other cases it may be desirable to direct the implementation of the present invention to symmetric session key distribution. In this case a temporary asymmetric crypto-key having a temporary private key portion and an associated temporary public key portion may be generated by the third party's computer. The third party's computer then encrypts the symmetric session crypto-key with the second private key portion of a user's long term crypto-key to form a first encrypted message. The third party's computer also encrypts the symmetric session crypto-key with the temporary public key portion to form a second encrypted message. A second user's computer obtains the symmetric session crypto-key by applying the temporary private key portion to decrypt the second encrypted message and encrypts the first encrypted message with the temporary private key portion to form a third encrypted message. The first user's computer obtains the symmetric session crypto-key by applying the temporary public key portion and the first user's first private key portion to decrypt the third encrypted message. Communications between the first user and the second user can now be encrypted and decrypted by the each user's computer with the symmetric session crypto-key.

Symmetric session distribution can alternatively be performed, in accordance with the present invention, by the third party's computer encrypting a first symmetric session key with the second private key portion of a first user's asymmetric crypto-key to form a first encrypted message and with a second symmetric session key to form a second encrypted message. The second user's computer obtains the first symmetric session key by applying the second symmetric session key to the second encrypted message. The second user's computer encrypts the first encrypted message with the temporary private key portion to form a third encrypted message. The first user's computer obtains the second symmetric session crypto-key by applying the second temporary key portion and the first private key portion of the first user crypto-key to the third encrypted message. Communication between the first user and the second user can now be encrypted and decrypted by the first and second users' computers with the second symmetric session crypto-key.

In either of the above cases, if appropriately authorized, the symmetric session crypto-key can be disclosed, by the third party, to a third user, such as the Federal Bureau of investigation (FBI), for eavesdropping on the encrypted communication.

#### BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a diagram of a conventional Kerberos authentication system.

FIG. 2 is a diagram of an authentication system according to the present invention.

FIGS. 3A and 3B are flow diagrams illustrating the steps for securing communications in accordance with the present invention.

FIG. 4 is a flow diagram illustrating the steps for forming joint signatures in accordance with the present invention.

FIG. 5 depicts a computer suitable for use as the client processor depicted in FIG. 2.

FIG. 6 is a exemplary block diagram of the computer depicted in FIG. 5.

FIG. 7 depicts a computer suitable for use as the authentication server depicted in FIG. 2.

FIG. 8 is a exemplary block diagram of the computer depicted in FIG. 7.

FIG. 9 depicts a computer suitable for use as the ticket granting server depicted in FIG. 2.

FIG. 10 is a exemplary block diagram of the computer depicted in FIG. 9.

FIG. 11 depicts a computer suitable for use as the service granting server depicted in FIG. 2.

FIG. 12 is a exemplary block diagram of the computer depicted in FIG. 11.

#### DESCRIPTION OF THE PREFERRED EMBODIMENT

For purposes of the following description it should be understood that [Message]Kc means the message is encrypted using a symmetric cryptosystem, such as DES, and crypto-key Kc. [Message]Dc means the RSA or some other modular exponentiation operator with the corresponding modulus Nc, i.e.  $[Message]D_c = (Message)^{D_c} \bmod N_c$ .

It should also be understood that the crypto-keys are created, as in any public-key cryptosystem, in accordance with the established policy. The creation and issuance of asymmetric crypto-key could for example, be performed by

an organization's Security Dept., perhaps the same organization that issues Photo Ids, using a terminal connected to a secure computer (e.g. a computer or processor with a tamper proof chip). A user c could access this terminal, enter her or his name, etc. This information is certified by a security officer, whose password or private key the computer knows. The computer then creates an RSA or other public-private key pair  $(E_c, N_c, D_c)$ , prompts the user for a password, which becomes  $D_c$ , the user c's portion of the RSA private key D. The computer computes  $D_{cy}$  which is the portion of the user's private key D which is stored in a secured database, referred to as the Yaksha database. As before,  $D = (D_c \times D_{cy}) \bmod \phi(N_c)$ . If the computer is also the authentication server acting as the certifying authority, it preferably computes  $[c, E_c, N_c]D_{ca}$  as that user's certificate. Any user can obtain user c's public key by applying the certifying authorities public key  $E_{ca}N_{ca}$  to the user's certificate. This is a simplification of the complex structure of an actual certificate but is sufficient for purposes of this discussion, Kent, S., "Privacy Enhancement for Internet Electronic Mail: Part II: certificate Based Key Management", INTERNET RFC 1422, February 1993.

Once smart cards are ubiquitous, the user-password becomes irrelevant and the security computer can download the user's (long) private key directly to a smart card. No method of key generation is critical to the functioning of the present invention, hence the above is only meant to be one possible scenario. Since the present invention is not vulnerable to the some of the attacks which conventional Kerberos systems are vulnerable to, the user's private key utilized in accordance with the present invention will have a longer useful life than in Kerberos.

Preferably, for every user, there exists a first private asymmetric crypto-key portion  $D_c$  known only to the user. It will be understood that a user may be a person or entity, a server or processor, or a system device such as a switch in a communications network. A second private crypto-key portion  $D_{cy}$  for every user is stored on a secured database, i.e. the Yaksha database. Certificates exist on a certifying authority's server, which is also referred to as the authentication server, and possibly on other servers and user processors, such as a ticket granting server, of the form  $[c, E_c, N_c]D_{ca}$ , and every user knows  $E_{ca}N_{ca}$  which is the certifying authority's public key. All other intermediate key generation information has been destroyed, preferably within the safe confines of the tamper proof chip used to generate the crypto-keys.

The present invention will now be described with reference to FIGS. 2, 3A, 3B and 4. FIG. 2 is an exemplary embodiment of the system and FIGS. 3A, 3B and 4 illustrate the steps performed by the various system components to provide enhanced system security and flexibility, in accordance with the present invention.

Referring first to FIG. 2, as shown the system includes a personal computer, workstation or other type of processor 110 operated by a user c. The processor 110 is connected to network 60 which is identical to the network shown in FIG. 1. The processor 110 can communicate with an authentication server 120, ticket granting server 140 and a service server 150 via the network. A YAKSHA database 130 is directly linked to the authentication server 120 and the ticket granting server 140. A processor or server 160 is also a user of the system and is connected to the system via the network 60. As noted above, each user, including each server, preferably has an asymmetric crypto-key assigned to it. The key is made up of a public-private key pair, the public portion of which is known to all users. The private portion of the key

15

is divided into a user portion  $D_c$  which is known only to the applicable user and a second portion  $D_{cy}$  which is stored on the YAKSHA database 130 and accessible only to the authentication server 120 and ticket granting server 140.

As will be described below, each of the messages transmitted in a conventional Kerberos type environment will have a corresponding message in the system and method of the present invention. As shown in FIG. 2, these messages are designated 1' through 6'.

Referring now to FIGS. 1, 2, 3A and 3B, in step 200 of FIG. 3A, a temporary RSA key pair  $D_{c\text{temp}}$ ,  $E_{c\text{temp}}$  and  $N_{c\text{temp}}$  are generated by processor 110. In Kerberos the initial  $as\_req$ ,  $as\_rep$  message 1 of FIG. 1 is:

Kerberos:  $as\_req: c, tgs, time-exp, n$  (1)

The corresponding message 1' of FIG. 2 is generated in step 210 of FIG. 3A and transmitted over network 60 by processor 110 of FIG. 2 is:

Yaksha:  $as\_req: c, tgs, time-exp, n, [TEMP-CERT]D_c, n]D_c$  (1')

In message 1' TEMP-CERT contains  $(c, E_{c\text{temp}}, N_{c\text{temp}}, expiry-time, \text{etc.})$  where  $E_{c\text{temp}}$ ,  $N_{c\text{temp}}$  is the public portion of the temporary RSA private-public key pair which the user  $c$  generated on processor 110, and expiry-time, is the time interval or period during which the temporary RSA key pair is valid. The time interval of validity will vary depending on the application but may nominally be set to correspond to one business day or some portion thereof. The TEMP-CERT is encrypted and hence "signed" with the user's portion of the long term private key,  $D_c$ . This structure is then concatenated with a random number string  $n$ , and again "encrypted" with  $D_c$ . Accordingly, an attacker who might later see TEMP-CERT, is prevented from seeing  $[TEMP-CERT]D_c$  and mounting a dictionary attack by taking guesses at  $D_c$  and checking if  $[TEMP-CERT]guess = [TEMP-CERT]D_c$ .

The authentication server 120 receives message 1' and performs  $[[[TEMP-CERT]D_c, n]D_c]D_{cy}]E_{cy}$ , as indicated in step 220 of FIG. 3A, to recover  $[TEMP-CERT]D_c$  and  $n$ . The server 120 retrieves  $D_{cy}$  from storage in Yaksha database 130. The server 120 then completes the signature on the temporary certificate by performing, in step 230 of FIG. 3A,  $[[[TEMP-CERT]D_c]D_{cy}]E_{cy}$  and successfully recovering the TEMP-CERT, the authentication server 120 can authenticate the user. It should also be noted that, in recovering TEMP-CERT, the server 120 recovers  $E_{c\text{temp}}$ ,  $N_{c\text{temp}}$ .

At this point in Kerberos the reply to the user, represented as message 2 in FIG. 1, is:

Kerberos:  $as\_rep: \{K_c, tgs, time-exp, n, \dots\} K_c, \{Tc, tgs\} K_{tgs}$  (2)

The corresponding message 2' generated by server 120 of FIG. 2 in steps 240 through 260 of FIG. 3A and transmitted via network 60 is:

Yaksha:  $as\_rep: \{K_c, tgs, time-exp, n, \dots\} E_{c\text{temp}}, [Tc, tgs] D_{tgsy}, [TEMP-CERT]D_c]D_{cy}$  (2')

It will be observed that the first two components of messages 2 and 2' are identical, except that, in message 2', the encryption is performed using modular exponentiation and using different keys. The third component of message 2' is the certificate signed by both the processor 110 and the server 120 verifying the authenticity of the temporary public-private key pair. The processor 110, after receipt of message 2', can "decrypt" the first part of the message 2'

16

using  $D_{c\text{temp}}$  which only it knows, to recover the usual Kerberos information as indicated in step 270 of FIG. 3A. The second portion of message 2' is the ticket granting ticket encrypted by server 120 with the portion  $D_{tgsy}$  of the ticket granting server 140 RSA private key which is retrieved from Yaksha database 130. It should be noted that the user private key  $D_c$  is not utilized in this transaction or communication other than in connection with messages 1' and 2'. Nor is the other portion of this private key, namely,  $D_{cy}$  used again, thus effectively preventing any dictionary attacks against  $D_c$ . However,  $D_c$  and  $D_{cy}$  make their "presence felt" since they have been used to encrypt and thus sign, the temporary public key,  $E_{c\text{temp}}$  and now the user, using processor 110, can "sign" messages with the corresponding private temporary key  $D_{c\text{temp}}$ , which is a regular full size RSA key invulnerable to password guessing attacks, without danger of revealing  $D_c$ . Further, a message can be sent securely to the user  $c$  encrypted under  $E_{c\text{temp}}$ , by any entity that sees the temporary certificate.

20 In Kerberos the request, in message 3 of FIG. 1, to the ticket granting server 140 takes the form:

Kerberos:  $tgs\_req: s, time-exp, n, \{Tc, tgs\} K_{tgs}, \{t, \dots\} K_c, tgs$  (3)

The only modifications to this message 3 made in message 3' of FIG. 2, are to attach the temporary certificate TEMP-CERT to the message 3', and to take the encrypted ticket from message 2', i.e.  $[Tc, tgs] D_{tgsy}$ , and to sign it using the temporary private key,  $D_{c\text{temp}}$ . The first modification allows the ticket granting server 140 to retrieve  $E_{c\text{temp}}$ ,  $N_{c\text{temp}}$  from TEMP-CERT and the second modification guarantees that a compromised authentication server 120 cannot generate a valid ticket for a "fake" user. The resulting message which is generated by the user using processor 110 and transmitted over network 60, and as indicated in steps 280-300 of FIG. 3A is:

Yaksha:  $tgs\_req: s, time-exp, n, \{Tc, tgs\} D_{tgsy} D_{c\text{temp}}, \{t, \dots\} K_c, tgs, [[TEMP-CERT]D_c]D_{cy}$  (3')

40 The ticket granting server 140 first recovers, for example from the user's permanent certificate, the user's public key portion  $E_c, N_c$ , uses this to recover the TEMP-CERT in step 310 of FIG. 3A. The server 140 then uses the temporary public key which is contained in the temporary certificate TEMP-CERT to retrieve  $[Tc, tgs] D_{tgsy}$  in step 320 of FIG. 3A, and then uses its private key  $D_{tgs}$  and public key  $E_{tgs}, N_{tgs}$  to recover the ticket  $Tc, tgs$  in step 330 of FIG. 3A.

At this point the ticket granting service 140 has authenticated the user, and the  $tgs\_rep$  message 4 of FIG. 1 is:

Kerberos:  $tgs\_rep: \{K_c, s, time-exp, n, \dots\} K_c, tgs, \{Tc, s\} K_s$  (4)

Message 4 could be utilized almost unchanged. For instance, simply by replacing  $\{Tc, s\} K_s$  with  $[Tc, s] D_{sy}$ . However, in such a case mutual authentication is not achieved and a compromised server 120 could spoof the user into believing it is talking to the server 140 when it is not. So the return message 4' of FIG. 2 is formed to contain proof of the authenticity of server 140. Since the user processor 110 already knows  $[Tc, tgs] D_{tgsy}$  which it received in step 280 of FIG. 3A as part of message 2', the server 140 completes the signature on this message in step 340 of FIG. 3A using its portion of its private key  $D_{tgs}$  and returns the result to the user via network 60. The  $tgs\_rep$  message 4' of FIG. 2, as indicated in steps 340-360 of FIG. 3A is:

Yaksha:  $tgs\_rep: \{K_c, s, time-exp, n, \dots\} K_c, tgs, \{Tc, s\} D_{sy}, [[Tc, tgs] D_{tgsy}] D_{tgs}$  (4')

In appropriate situations, a message 9' requesting access to the session key Kc may be forwarded to the server 140 via the network 60 from the law enforcement processor or server 160. Assuming appropriate authority can be verified, such as by using the previously described authentication process, the server 140 forwards, in message 10', the session key to the processor 160 thus facilitating a legal wiretap being established on communications between the processor 110 and the server 150. It will be noted that this allows eavesdropping by the law enforcement processor 160 of communications over the network 60 between processor 110 and server 150 during the session, without disclosing to the law enforcement processor 160 the long term private keys of either user. Thus, the long term security of the system has not been jeopardized.

The client via processor 110 retrieves message 4' and, for example using the server 140 permanent certificate, recovers the public key Btgs.Ntgs for server 140 and verifies in step 370 of FIG. 3A that [[Tc.tgs]Dtgsy]Dtgs is the signature on a valid ticket Tc.tgs.

The Kerberos messages 3 and 5 are fundamentally identical, the former being a special type of request to a server. Similarly, message 5' of FIG. 2 generated by user processor 110 is step 380-400 of FIG. 3B and communicated via network 60 is identical to message 3' of FIG. 2. The messages 5 of FIG. 1 and 5' of FIG. 2 are:

Kerberos:ap\_req: {ts, ck, ... } Kc, s {Tc, s} Ks (5)

Yaksha: {ts, ck, ... } Kc, s, [[Tc, s]Dey] Dc, temp, [[TEMP-CERT] Dc] Dey (5')

It will be noted that message 5' has a general form which is similar to the message 3' generated in steps 280-300 of FIG. 3A. Thus, server 150 will perform steps similar to those shown in steps 310-330 of FIG. 3A in corresponding steps 410-430 of FIG. 3B. These steps will not be further described to avoid unnecessary duplication.

Mutual authentication is again mandated and the server 150 must prove its knowledge of its long term private key Ds. As in message 4' of FIG. 2 this is achieved by the server 150 sending back in message 6' of FIG. 2, via network 60, the service ticket Tc, s, with the signature completed. This allows the authenticity of server 150 to be verified by processor 110, in step 450 of FIG. 3B, using the long term public key E, N, of server 150. Consequently messages 6 of FIG. 1 and 6' of FIG. 2, the later of which is generated by server 150 in step 440, are:

Kerberos:ap\_rep: {ts} Kc, s (6)

Yaksha: ap\_rep: [[Tc, s]Dey] Ds (6')

It will be observed that both messages 4' and 6' are modified versions of messages 4 and 6, respectively, which facilitate mutual authentication, without having to trust the server 120 or server 140.

The forming of joint signatures in accordance with the present invention will now be described with reference to FIGS. 2 and 4. FIG. 4 illustrates the steps performed by the various system components to form a joint signature on a message and thus provide for non-repudiation.

Kerberos does not perform joint signatures, so the following messages do not have a Kerberos counterpart. Rather these messages are a modification of the signature protocol described in Page J. and R. Plant, "A Secure Joint Signature and Key Exchange System", removed for anonymous review. The modified messages significantly improve the security of the system against a potential dictionary attack. The messages are:

Yaksha: sign\_req: c, [[H, ts] Dc, n] Dc, temp, [[TEMP-CERT] Dc, n] Dc (7)

Yaksha: sign\_rep: [[[[H, ts] Dc] Dcy, n] Ec, temp (8)

In forming joint signatures, the temporary asymmetric crypto-key, generated by processor 110 of FIG. 2, as described in step 200 of FIG. 3A, having a private temporary key portion Dc, temp and an associated public temporary key portion Ec, temp Nc, temp, is utilized. The public temporary key portion Ec, temp Nc, temp is encrypted, as part of the temporary certificate TEMP-CERT, with the long term private key portion Dc of the user c to form a first encrypted message as in step 210 of FIG. 3A. The public temporary key portion Ec, temp Nc, temp is obtained by, for example, the authentication server 120 of FIG. 2, by applying the second private key portion Dcy and public key portion Ecy Ncy of user c to the first encrypted message, as in step 220 of FIG. 3A, to authenticate user c to the authentication server 120. These steps have not been reiterated in FIG. 4. Thus, the same temporary public-private key pair are used to perform both mutual authentication, as described with reference to FIGS. 3A and 3B, and joint signatures between the user and the server.

Preferably, the user c using processor 110, in step 510 of FIG. 4, signs, with private key portion Dc known only to user c, a hash message H, concatenated, optionally, with a time stamp ts to add redundancy to the message, although in practice, a signature would often have some well defined format and the time stamp may not be necessary. In step 520 of FIG. 4, the user, via processor 110, concatenates this signed message with a random number string n to prevent dictionary attacks of the form [H, ts] guess, and then signs again with the private temporary key portion Dc, temp. Message 7' is formed by adding [[TEMP-CERT] Dc, n] Dc which includes the public temporary key portion Ec, temp Nc, temp as described in step 210 of FIG. 3A. Processor 110 transmits the message 7' over the network 60. On receipt of message 7', the authentication server 120 first unlocks the TEMP-CERT, just as in step 220 of FIG. 3A, and recovers the temporary public key Ec, temp Nc, temp. Server 120 uses, in step 530 of FIG. 4, the public temporary key portion Ec, temp to recover [H, ts] Dc and n. The server 120 then, in step 540, computes [[H, ts] Dc] Dcy, having retrieved Dcy from Yaksha database 130, which serves as the signature of server 120 on the hash message and time stamp. Server 120 then, in step 550 of FIG. 4, concatenates the jointly signed hash message and time stamp with a random number string n, encrypts the jointly signed message and n using the temporary public key Ec, temp and transmits this encrypted message via network 60. The user can recover the jointly signed hash message and time stamp in step 560 by applying, via processor 110, the temporary private key Dc, temp known only to user c, and then verify the authenticity of the signature using its long-term public key Ec, Nc.

FIGS. 5-12 depict exemplary computers suitable for use as the client processor 10, and the servers 120, 140 or 150 shown in FIG. 2. The computers are preferably commercially available personal computers or high-powered work stations. Each computer's processor could, for example, be a Pentium™ processor. The computers are depicted as having similar hardware configurations, although this is not necessarily required. For example, as will be well understood by the skilled artisan, it may be desirable for components of the respective computers to have attributes such memory storage capacity, data transmission rates and/or processing speeds which differ. Virtually any commercially available keyboard, mouse and monitor can be utilized. A high-speed network interface, including a high-speed modem, is pre-

ferred although not mandatory. One or more of the computers, if desired, could also or alternatively include other components (not shown), such as an optical storage medium or may exclude depicted components.

Each of the computers differ in their respective computer programming instructions. Hence, the functionality of each of the computers described with reference to FIGS. 5-12 varies from that of the other computers due to the programming instructions which drive its operation. It will be recognized that only routine programming is required to implement the disclosed instructions on the described computers such that the computers are driven by the programming to operate in accordance with the invention.

To avoid unnecessary duplication common features of the computers depicted in FIGS. 5-12 will be described only with reference to FIGS. 5 and 6. It should be understood that the corresponding components of the computers depicted in FIGS. 7-12 will be similar. Further, since the computer components and configurations are conventional, routine operations performed by the depicted components will generally not be described, such operations being well understood in the art.

Preferably, each of the computers stores its unique programming instructions on its ROM or hard disk. Portions of long term crypto-keys are preferably stored in each computer on the hard disk. Portions of temporary crypto-keys, session keys and other short term data which is required to be processed or otherwise utilized more than once is preferably stored on the RAM. The computer 600 or 600', i.e. either of the computers which serve as the authentication server 120 or ticket granting server 140 of FIG. 2, could if desired include the Kerberos database 130 of FIG. 2 stored preferably on its hard disk.

Referring now to FIG. 5, the computer 600 includes a main unit 610 with slots 611, 612 and 613, respectively provided for loading programming or data from a floppy disc 726a, CD 728a and smart card 729a onto the computer 600. The computer 600 also includes a keyboard 630 and mouse 640 which serve as user input devices. A monitor display 620 is also provided to visually communicate information to the user.

As depicted in FIG. 6, The computer 600 has a main processor 700 which is interconnected via bus 710 with various storage devices including RAM 720, ROM 722 and hard drive 724 with hard disk 724a, all of which serve as a storage medium on which computer programming or data can be stored for access by the processor 700. The main processor 700 is interconnected via bus 710 with various other storage devices such as the floppy disc drive 726, the CD drive 728 and the card reader 729 which are capable of being controlled by drive controller 750 to read computer programming or data stored on a floppy disc 726a, CD 728a or smart card 729a when inserted into the appropriate slot 611, 612 or 613 in the unit 610. By accessing the stored computer programming the processor 700 is driven to operate in accordance with the present invention.

The processor 700 is also operatively connected to the keyboard 630 and/or mouse 640, via input interface 730. The display monitor 620 is interconnected to the processor 700, via display interface 740, to facilitate the display of information to the user. The network interface 760 is provided to interconnect the processor 700 to the network 60 depicted in FIG. 2 and accordingly allow communications between the computer 600 and other network devices. Since the computer 600 serves as the client processor 110 of FIG. 2, the network interface allows communications with network servers 120, 140 and 150 of FIG. 2.

The inter-operation of the various components of the computers depicted in FIGS. 5-12 in implementing the steps described above with reference to FIGS. 2-4 will now be described. Referring first to FIGS. 5 and 6, in order for the first user to communicate with another user on the Yaksha system, the user enters a command using the keyboard 630 or mouse 640, responsive to which the computer programming stored, for example on the ROM 722 drives the processor 700 to generate a temporary RSA asymmetric crypto-key pair Dc temp, Ec temp and Nc temp, as described in step 200 of FIG. 3A.

In accordance with the stored programming instructions, the processor 700 next prompts the user to enter, via keyboard 630 the user's password D<sub>c</sub>, which is preferably an 8 to 12 character segment of the user's RSA private key, by displaying an inquiry on the monitor 620. Alternatively, the user's password D<sub>c</sub> could be stored on smart card 729a and entered via card reader 729, thereby allowing retrieval of the password D<sub>c</sub> by the processor 700. The use of the smart card 729a for storing the password D<sub>c</sub> is particularly beneficial if the password is longer than 12 characters or is of a type not easily remembered by the user. The processor 700 could, if desired, be instructed to temporarily store the password D<sub>c</sub> in the RAM 720. Processor 700 is next driven by the stored programming to generate encrypted message 1' of FIG. 2, as described in step 210 of FIG. 3A. The network interface 760 is directed by the processor 700 to transmit the encrypted message over network 60. As discussed above, in message 1' TEMP-CERT contains (c, Ec temp, Nc temp, expiry-time, etc.). Ec temp, Nc temp is the public portion of the temporary RSA private-public key pair. The expiry-time is the time interval or period during which the temporary RSA key pair is valid. In encrypted message 1' generated by the processor 700, the TEMP-CERT is encrypted and hence "signed" with the user's portion of the long term private key, Dc. The processor 700 also concatenates this structure with a random number string n, and further encrypts the message with Dc, all in accordance with instructions from the programming stored on the ROM 722. As discussed previously, this shields [TEMP-CERT]Dc from dictionary attack, thereby improving the crypto-system security.

Referring now to FIGS. 7 and 8, the computer 600' serves as the authentication server 120. Encrypted message 1' is received by the network interface 760' and conveyed to processor 700'. In accordance with instructions from programming stored on ROM 722', the processor 700' retrieves Dcy, Ecy from storage, for example on hard disk 724a', and performs [([TEMP-CERT]Dc, n)Dc]Dcy[Ecy], as discussed in step 220 of FIG. 3A. It will be noted that as described, hard disk 724a' stores D<sub>c</sub>, and hence the Yaksha database, although this is not a necessary requirement as the Yaksha database could be stored on or separate from any of the computers described in FIGS. 5-12. As indicated above, the processor 700' recovers [TEMP-CERT]Dc and n. The processor 700' then completes the signature on the temporary certificate by performing ([TEMP-CERT]Dc]Dcy, as discussed in step 230 of FIG. 3A. By performing [([TEMP-CERT]Dc]Dcy]Ecy and successfully recovering the TEMP-CERT, the user has been authenticated. Further, in recovering TEMP-CERT the processor 700' recovers Ec temp, Nc temp.

As described in steps 240 through 260 of FIG. 3A, the processor 700', pursuant to instructions from the programming stored on ROM 722' next retrieves D<sub>cy</sub> from the Yaksha database stored on hard disk 724a' and generates message 2'. The third component of message 2' includes the certificate signed by both the user processor 700 and the

authentication server processor 700', and accordingly provides verification of the authenticity of the temporary public-private key pair. In accordance with the stored programming, the processor 700' directs the network interface 760' to transmit message 2' to computer 600 via network 60.

The processor 700 receives message 2' via network interface 760, and, in accordance with instructions from the programming stored on ROM 722 retrieves Dc.temp from storage on, for example, the RAM 720 and decrypts the first part of the message 2' by applying Dc.temp, which only it knows, to recover the usual Kerberos information as indicated in step 270 of FIG. 3A. The second portion of message 2' is the ticket granting ticket encrypted by processor 700' with the portion D<sub>key</sub> of the RSA private key for ticket granting server 140. As discussed earlier, the user private key Dc, Dcy are not utilized in this transaction or communication other than in connection with messages 1' and 2', thus effectively preventing any dictionary attacks against the user's private key. However, Dc and Dcy form the signatures on the temporary public key Ectemp. Accordingly, the user processor 700, can, without danger of revealing Dc, "sign" messages using the private temporary key Dctemp, which is a regular full size RSA key invulnerable to password guessing attacks. Additionally, any entity with access to the temporary certificate can send messages securely to the user c encrypted under Ec.temp.

The processor 700, in accordance with instructions from the programming stored on ROM 722, again retrieves Dc.temp and generates message 3' of FIG. 2, as indicated in steps 280-300 of FIG. 3A. Message 3' includes the temporary certificate TEMP-CERT. This allows the ticket granting server 140 to retrieve Ec.temp, Nc.temp from TEMP-CERT. Message 3' also includes the encrypted ticket from message 2', i.e. [Tc.tgs]Dtgsy. This guarantees that a compromised authentication server 120 does not generate a valid ticket for a "fake" user. Message 3' is also signed using the temporary private key, Dc.temp. In accordance with further instructions from the programming stored on ROM 722, the processor 700 directs the network interface 760 to forward message 3' to the ticket granting server 140 of FIG. 2 via network 60.

Referring to FIGS. 9 and 10, the ticket granting server computer 600" receives message 3' via network interface 760". In accordance with instructions from programming stored on ROM 722", processor 700" recovers, for example from the user's permanent certificate, the user's public key portion Ec.Nc., and uses this to recover the TEMP-CERT, as described in step 310 of FIG. 3A. As instructed by the stored programming, the processor 700" then uses the temporary public key from the temporary certificate TEMP-CERT to retrieve [Tc.tgs]Dtgsy, as described in step 320 of FIG. 3A. The processor, pursuant to further programming instructions next retrieves from storage on, for example hard disk 724a", and uses its private key Dgts and public key Etgs.Ntgs to recover the ticket Tc.tgs, as described in step 330 of FIG. 3A. As previously discussed, the ticket granting service 140 has now authenticated the user.

The stored programming on ROM 722" now instructs the processor 700" to generate the return message 4' of FIG. 2 in a manner which includes proof of the authenticity of server 140. In this way a compromised server 120 cannot spoof the user into believing it is talking to the server 140 when it is not. Since the user processor 700 already knows [Tc.tgs]Dtgsy which it received as part of message 2', as indicated in step 280 of FIG. 3A, the processor 700" completes the signature on this message using its portion of its private key Dgts and generates the tgs\_rep message 4' of

FIG. 2, as indicated in steps 340-360 of FIG. 3A. The stored programming now instructs the processor 700" to direct the network interface 760" to forward message 4' to the user via network 60.

As has been previously discussed, in appropriate situations, a message requesting access to the session key Kc may be forwarded to the ticket granting server processor 700" via the network 60 from the law enforcement processor or server 160 of FIG. 2. Assuming appropriate authority is verified, perhaps by using the previously described authentication process, the processor 700", pursuant to instructions from the stored programming, forwards a message including the session key to the processor 160 thus facilitating a legal wiretap being established on communications between the user computer 600 and the service server computer 600", which will be described in detail below. Accordingly, legal eavesdropping by the law enforcement processor 160 of communications over the network 60 during a session are facilitated, without disclosing the long term private keys of either user to the law enforcement processor 160 and jeopardizing long term system security.

The computer 600 retrieves message 4' via network interface 760. In accordance with the programming instructions stored on ROM 722, and for example using the server 140 permanent certificate, processor 700 recovers the public key Etgs.Ntgs for server 140 from hard disk 724a and verifies, as described in step 370 of FIG. 3A, that [[Tc.tgs]Dtgsy]Dtgs is the signature on a valid ticket Tc.tgs.

Message 5' of FIG. 2 is generated, as described in steps 380-400 of FIG. 3B, by the user's computer 600. This message is similar to message 3' of FIG. 2 which, as described above, is likewise generated by the user's computer 600. Accordingly, the generation of message 5' by computer 600 will not be further described.

Service server computer 600" will decrypt message 5' by performing steps similar to those performed by ticket granting server computer 600" and as described in steps 310-330 of FIG. 3A. Accordingly corresponding steps 410-430 of FIG. 3B will not be further described.

Referring to FIGS. 11 and 12, as with message 4' of FIG. 2, mutual authentication is achieved by service server computer 600" proving knowledge of its long term private key Ds. Hence, in accordance with instructions from programming stored on ROM 722", processor 700" retrieves Ds, from for example hard disk 724a", and generates, as described in step 440, message 6' of FIG. 2, which includes the service ticket Tc.s with the signature completed. Pursuant to the stored programming instructions, the processor 700" directs the network interface 760" to transmit message 6' via network 60 back to the computer 600. The authenticity of the service server 150 is verified by the processor 700, in accordance with programming instructions stored on ROM 722 and as described in step 450 of FIG. 3B, by retrieving from storage on, for example, hard disk 724a and applying the long term public key E<sub>N</sub>, of server 150. Both messages 4' and 6' facilitate mutual authentication, without having to trust the server 120 or server 140.

The forming of joint signatures in accordance with the present invention will now be described with reference to FIGS. 2, and 4-8. FIG. 4 illustrates the steps performed to form a joint signature on a message and thus provide for non-repudiation.

In forming joint signatures, computers 600 and 600' perform steps 200-220 of FIG. 3A as previously described. Therefore performance of these steps will not be reiterated. Pursuant to instructions from the programming stored on ROM 722, processor 700, and signs a hash message H.

optionally concatenated with a time stamp  $t_s$  to add redundancy to the message, with  $D_c$  as discussed in step 510 of FIG. 4. In accordance with the stored programming instructions, processor 700 next concatenates this signed message with a random number string  $n$  to prevent dictionary attacks of the form  $[H, t_s]_{\text{guess}}$ , and then again signs the message with the private temporary key portion  $D_{c, \text{temp}}$ , as described in step 520 of FIG. 4. As further instructed by the stored programming, the processor 700 generates message 7 by adding  $[[\text{TEMP-CERT}]D_{c, n}]D_c$  which includes the public temporary key portion  $E_{c, \text{temp}} N_{c, \text{temp}}$  as described in step 210 of FIG. 3A. Processor 700 is then instructed by the stored programming to direct the network interface 760 to transmit message 7 over the network 60.

Message 7 is received by authentication server processor 700' via network interface 760'. As instructed by the programming stored on the ROM 722', the processor 700' first unlocks the TEMP-CERT, just as described above in the performance of step 220 of FIG. 3A by processor 700'. Processor 700' thereby recovers the temporary public key  $E_{c, \text{temp}} N_{c, \text{temp}}$ .

In accordance with further instructions from the stored programming, processor 700' uses the public temporary key portion  $E_{c, \text{temp}}$  to recover  $[H, t_s]D_c$  and  $n$ , as described in step 530 of FIG. 4. The processor 700' is now directed by the stored programming to compute  $[[H, t_s]D_c]D_{c, y}$ , as described in step 540, having retrieved  $D_{c, y}$  from Yaksha database stored on hard disk 724a'. The addition of  $D_{c, y}$  serves as the signature of server 120 on the time stamped hash message. In accordance with further instructions from the stored programming, the processor 700' concatenates the jointly signed hash message and time stamp with a random number string  $n$  and encrypts the jointly signed message and  $n$  using the temporary public key  $E_{c, \text{temp}}$ , as discussed in step 550 of FIG. 4. The stored programming now instructs the processor 700' to direct the network interface 760' to transmit the encrypted message via network 60.

The user processor 700 receives the encrypted message via network interface 760 and, in accordance with instructions from the programming stored on ROM 722, recovers the jointly signed hash message and time stamp as described in step 560 by retrieving from RAM 720 and applying the temporary private key  $D_{c, \text{temp}}$ . The processor 700 is then instructed to retrieve its long-term public key  $E_{c, N_c}$  from the hard disk 724a and verify the authenticity of the signature using its long-term public key  $E_{c, N_c}$ .

In accordance with the present invention, in order for a user to authenticate itself to the certification or authentication server, and visa versa, the user must reveal knowledge of  $D_c$  to the server, and the server must reveal knowledge of  $D_{c, y}$  to the user. Further, when a user receives a ticket from another user it requires proof that the authentication server has vouched for the ticket, and further, unlike in conventional Kerberos, requires proof that the user has requested the ticket, i.e. it trusts neither the user nor the authentication server individually, but it trusts the message if both vouch for it. Similarly, the mutual authentication response to the initiating user requires a message vouched for by both the other user and the authentication server.

However, like in Kerberos, the user's private key  $D_c$  is stored for no more than a short, i.e. the minimum, period of time. Hence a temporary RSA private-public key pair is generated on the fly, that is on-line in real time, and the user and the authentication server collaborate to sign the public portion of this temporary key to create a temporary certificate that is valid for, say, eight hours. It will be noted that the authenticity of this temporary pair is verified using the long

term public key. Further, the authentication server never sees the private key portion of the temporary pair. This entire artifice may disappear once smart cards are ubiquitous, and computations involving a user's private key occur inside the smart card connected to the computer. Also, legal wiretaps can be established without the disclosure of users' long term private keys. All of the above is achieved, in accordance with the present invention, with minimal changes to the conventional Kerberos protocol. Preferably, users can retrieve certificates from a particular server other than the authentication server. Alternately, the appropriate certificates could be attached to messages from the authentication server.

As described above, the present invention provides a programmed computer and computer programming for securing communications in which the compromise of a central database, such as the secured database in a conventional Kerberos system, will not be catastrophic to the overall system security. The computer and programming of the present invention also make the crypto-system less vulnerable to dictionary attacks and provide a way for one user to authenticate itself to another user. The described computer and programming facilitate digital signatures being placed on a message and thereby provide for non-repudiation. Additionally the computer and programming of the present invention can be implemented to enhance security in conventional Kerberos systems with minimum changes to the standard Kerberos protocol and are compatible with the use of "smart cards". Finally, the described computer and programming allow the reuse of an authentication infrastructure for digital signatures.

It will also be recognized by those skilled in the art that various features and aspects of the above described invention may be used individually or jointly. For example aspects of the invention relating to authentication, joint signatures and session key exchange may be implemented together or individually as may be desired for a particular application. Additionally, although in the description of the preferred embodiments of the invention the public key portion of the user's public/private key pair is applied by specific users, it will be understood that the application of the public key portion of the user's public/private key pair in, for example, authentication and key exchange, can often be performed by either user. Finally, although the preferred embodiments are described in the context of a Kerberos type system, those skilled in the art will recognize that the present invention can be beneficially utilized in any crypto-system having a secure central data base in which to store crypto-keys.

What is claimed:

1. An article of manufacture for securing communications between users of a crypto-system having a plurality of users, each of said plurality of users having an associated asymmetric crypto-key with a public key portion and a corresponding private key portion, each public key portion being accessible to the plurality of system users, each private key portion having a first private key portion known only to the associated user and a corresponding second private key portion, comprising:

computer readable storage medium; and

computer programming stored on said storage medium; wherein said stored computer programming is configured to be readable from said computer readable storage medium by a computer and thereby cause said computer to operate so as to:

generate a temporary asymmetric crypto-key having a first temporary key portion and an associated second temporary key portion;



encrypt said second temporary key portion with the first private key portion of a first user crypto-key associated with a first user to form a first encrypted message;

direct issuance of said first encrypted message to a second user having access to the second private key portion of the first user crypto-key, wherein the second temporary key portion is obtainable by application of the second private key portion of the first user crypto-key to the first encrypted message to thereby authenticate the first user to the second user; and

apply the public key portion of the first user crypto-key to decrypt a second encrypted message, which includes the first encrypted message encrypted with the second private key portion of the first user crypto-key, to thereby authenticate the second user to the first user.

2. An article of manufacture according to claim 1, wherein said stored computer programming is configured to be readable from said computer readable storage medium by the computer to thereby cause said computer to operate so as to:

apply the first temporary key portion to decrypt a third encrypted message, which includes a first symmetric session crypto-key encrypted with the second temporary key portion, to obtain a first symmetric session crypto-key;

apply the first temporary key portion to encrypt a fourth encrypted message, which includes the first symmetric session crypto-key encrypted with the second private key portion of a third user, to form a fifth encrypted message; and

direct issuance of said fifth encrypted message to said third user, wherein the first symmetric session crypto-key is obtainable by application of the first private key portion of the third user crypto-key and the second temporary key portion to decrypt the fifth encrypted message and thereby authenticate the first user and the second user to the third user.

3. An article of manufacture according to claim 2, wherein said stored computer programming is configured to be readable from said computer readable storage medium by the computer to thereby cause said computer to operate so as to apply said first symmetric session key to encrypt a first communication to the third user, or to decrypt a second communication from the third user.

4. An article of manufacture according to claim 2, wherein said stored computer programming is configured to be readable from said computer readable storage medium by the computer and thereby cause said computer to operate so as to:

apply the first symmetric session crypto-key to decrypt a sixth encrypted message, which includes a second symmetric session crypto-key encrypted with said first symmetric session crypto-key, to obtain the second symmetric session crypto-key;

apply the public key portion of the third user crypto-key to decrypt a seventh encrypted message, including the first symmetric session crypto-key encrypted with the first and the second private key portions of the third user, to obtain the first symmetric session crypto-key and thereby authenticate the third user to the first user;

apply the first temporary key portion to encrypt an eighth encrypted message, which includes the second symmetric session crypto-key encrypted with the second

private key portion of a fourth user crypto-key associated with a fourth user, to form a ninth encrypted message;

direct issuance of said ninth encrypted message to the fourth user, wherein the second symmetric session crypto-key is obtainable by application of the second temporary key portion and the first private key portion of the fourth user crypto-key to decrypt the ninth encrypted message and thereby authenticate the first user and the second user to the fourth user.

5. An article of manufacture according to claim 4, wherein said stored computer programming is configured to be readable from said computer readable storage medium by the computer and thereby cause said computer to operate so as to decrypt a tenth encrypted message, which includes the second symmetric session crypto-key encrypted with the first and the second private key portions of the fourth user crypto-key, by application of the public key portion of the fourth user crypto-key to the tenth encrypted message and thereby authenticate the fourth user to the first user.

6. A programmed computer for securing communications between users of a crypto-system having a plurality of users, each of said plurality of users having an associated asymmetric crypto-key with a public key portion and a corresponding private key portion, each public key portion being accessible to the plurality of system users, each private key portion having a first private key portion known only to the associated user and a corresponding second private key portion, comprising:

a processor for generating a temporary asymmetric crypto-key having a first temporary key portion and an associated second temporary key portion, for encrypting said second temporary key portion with the first private key portion of a first user crypto-key associated with a first user to form a first encrypted message, for directing issuance of said first encrypted message to a second user having access to the second private key portion of the first user crypto-key, and for applying the public key portion of the first user crypto-key to decrypt a second encrypted message, which includes the first encrypted message encrypted with the second private key portion of the first user crypto-key, to thereby authenticate the second user to the first user; and

storage medium for storing the first temporary key portion, and the public key portion of the first user crypto-key.

7. A programmed computer according to claim 6, wherein:

said processor is adapted (i) to apply the first temporary key portion to decrypt a third encrypted message, which includes a first symmetric session crypto-key encrypted with the second temporary key portion, to obtain the first symmetric session crypto-key, (ii) to apply the first temporary key portion to encrypt a fourth encrypted message, which includes the first symmetric session crypto-key encrypted with the second private key portion of a third user crypto-key associated with a third user, to form a fifth encrypted message, and (iii) to direct the issuance of said fifth encrypted message to said third user;

said storage medium is adapted to store said first symmetric session crypto-key; and

the first symmetric session crypto-key is obtainable by application of the first private key portion of the third user crypto-key and the second temporary key portion



27

to decrypt the fifth encrypted message and thereby authenticate the first user and the second user to the third user.

8. A programmed computer according to claim 7, wherein said processor is adapted to encrypt a message to or decrypt a message from the third user with said first symmetric session key.

9. A programmed computer according to claim 7, wherein:

said processor is adapted (i) to apply the first symmetric session crypto-key to decrypt a sixth encrypted message, which includes a second symmetric session crypto-key encrypted with said first symmetric session crypto-key, to obtain the second symmetric session crypto-key, (ii) to apply the public key portion of the third user crypto-key to decrypt a seventh encrypted message, which includes the first symmetric session crypto-key encrypted with the first and the second private key portions of the third user, to obtain the first symmetric session crypto-key and thereby authenticate the third user to the first user, (iii) to apply the first temporary key portion to encrypt an eighth encrypted message, which includes the second symmetric session crypto-key encrypted with the second private key portion of a fourth user crypto-key associated with a fourth user, to form a ninth encrypted message, and (iv) to direct issuance of said ninth encrypted message to the fourth user;

said storage medium is adapted to store said second symmetric session crypto-key, and the public key portion of the third user crypto-key; and

the second symmetric session crypto-key is obtainable by application of the second temporary key portion and first private key portion of the fourth user crypto-key to decrypt the ninth encrypted message and thereby authenticate the first user and the third user to the fourth user.

10. A programmed computer according to claim 9, wherein said processor is adapted to apply the public key portion of the fourth user crypto-key to decrypt a tenth encrypted message, which includes the second symmetric session crypto-key encrypted with the first and the second private key portions of the fourth user crypto-key, to obtain the second symmetric session crypto-key and thereby authenticate the fourth user to the first user.

11. An article of manufacture for jointly signing communications between users in a crypto-system having a plurality of system users, each said user having an associated asymmetric crypto-key with a public key portion and a corresponding private key portion, each public key portion being accessible to the plurality of system users, each private key portion having a first private key portion known only to the associated user and a corresponding second private key portion, comprising:

computer readable storage medium; and  
computer programming stored on said storage medium; wherein said stored computer programming is configured to be readable from said computer readable storage medium by a computer and thereby cause said computer to operate so as to:

generate a temporary asymmetric crypto-key having a first temporary key portion and an associated second temporary key portion;

apply to the first private key portion of a first user crypto-key associated with a first user to encrypt said second temporary key portion to form a first encrypted message;

28

direct issuance of the first encrypted message to a second user having access to the second private key portion of the first user crypto-key;

apply the first private key portion of the first user crypto-key to encrypt a hash message to form a second encrypted message, and thereby place a signature of the first user on the hash message;

apply said first temporary key portion to encrypt said second encrypted message to form a third encrypted message;

direct issuance of the third encrypted message to the second user; and

apply the public key portion of the first user crypto-key to decrypt a fourth encrypted message, which includes the second encrypted message encrypted with the second private key portion of the first user crypto-key, to obtain the hash message to thereby verify the joint signatures of the first and the second users on the hash message;

wherein, the second temporary key portion is obtainable by applying the second private key portion of the first user crypto-key to decrypt the first encrypted message and thereby authenticate the first user to the second user, and the second encrypted message is obtainable by applying said second temporary key portion to decrypt the third encrypted message.

12. A programmed computer for jointly signing communications between users of a crypto-system having a plurality of users, each of said plurality of users having an associated asymmetric crypto-key with a public key portion and a corresponding private key portion, each public key portion being accessible to the plurality of system users, each private key portion having a first private key portion known only to the associated user and a corresponding second private key portion, comprising:

a processor (i) for generating a temporary asymmetric crypto-key having a first temporary key portion and an associated second temporary key portion, (ii) for applying the first private key portion of a first user crypto-key associated with a first user to encrypt said second temporary key portion to form a first encrypted message, (iii) for directing issuance of the first encrypted message to a second user having access to the second private key portion of the first user crypto-key, (iv) for applying the first private key portion of the first user crypto-key to encrypt a hash message to form a second encrypted message, and thereby place a signature of the first user on the hash message, (v) for applying said first temporary key portion to encrypt said second encrypted message to form a third encrypted message, (vi) for directing to issuance of the third encrypted message to the second user, and (vii) for applying the public key portion of the first user crypto-key to decrypt a fourth encrypted message, which includes the second encrypted message encrypted with the second private key portion of the first user crypto-key, to obtain the hash message and thereby verify the joint signatures of the first and the second users on the hash message; and

storage medium for storing the first temporary key portion, and the public key portion of the first user crypto-key;

wherein, the second temporary key portion is obtainable by applying the second private key portion of the first user crypto-key to decrypt the first encrypted message and thereby authenticate the first user to the second user, and the second encrypted message is obtainable

by applying said second temporary key portion to decrypt the third encrypted message.

13. An article of manufacture, for authenticating users of a system having a plurality of system users, each said user having an associated asymmetric crypto-key with a public key portion and a corresponding private key portion, each public key portion being accessible to the plurality of system users, each private key portion having a first private key portion known only to the associated user and a corresponding second private key portion known only to a trusted third party, comprising:

computer readable storage medium; and  
computer programming stored on said storage medium;  
wherein said stored computer programming is configured to be readable from said computer readable storage medium by a computer and thereby cause said computer to operate so as to:  
apply the first private key portion of a first user crypto-key associated with a first user to encrypt a first message to form a first encrypted message;  
direct issuance of said first encrypted message to said trusted third party; and  
apply the public key portion of the first user crypto-key to decrypt to a second encrypted message, which includes the first encrypted message encrypted with the second private key portion of the first user crypto-key, to obtain the first message and thereby authenticate the trusted third party to the first user;  
wherein the first message is obtainable by applying the second private key portion of the first user crypto-key to decrypt the first encrypted message and thereby authenticate the first user to the trusted third party.

14. An article of manufacture according to claim 13, wherein the first message is obtainable by applying the public key portion of the first user crypto-key to decrypt the second encrypted message and thereby authenticate the first user to a second user.

15. A programmed computer for authenticating users of a system having a plurality of system users, each said user having an associated asymmetric crypto-key with a public key portion and a corresponding private key portion, each public key portion being accessible to the plurality of system users, each private key portion having a first private key portion known only to the associated user and a corresponding second private key portion known only to a trusted third party, comprising:

a processor (i) for applying the first private key portion of a first user crypto-key associated with a first user to encrypt a first message to form a first encrypted message, (ii) for directing issuance of said first encrypted message to said trusted third party, and (iii) for applying the public key portion of the first user crypto-key to decrypt a second encrypted message, which includes the first encrypted message encrypted with the second private key portion of the first user crypto-key, to obtain the first message and thereby authenticate the trusted third party to the first user; and  
storage medium for storing the public key portion of the first user crypto-key and the first message;

wherein the first message is obtainable by applying the second private key portion of the first user crypto-key to decrypt the first encrypted message and thereby authenticate the first user to the trusted third party.

16. A programmed computer according to claim 15, wherein the first message is obtainable by applying the

public key portion of the first user crypto-key to decrypt the second encrypted message and thereby authenticate the first user to a second user.

17. An article of manufacture for securing communications between users of a system having a plurality of system users, each said user having an associated asymmetric crypto-key with a public key portion and a corresponding private key portion, each public key portion being accessible to the plurality of system users, each private key portion having a first private key portion known only to the associated user and a corresponding second private key portion known only to a trusted third party, comprising:

computer readable storage medium; and  
computer programming stored on said storage medium;  
wherein said stored computer programming is configured to be readable from said computer readable storage medium by a computer and thereby cause said computer to operate so as to:

generate a temporary asymmetric crypto-key having a first temporary key portion and an associated second temporary key portion;

apply the first private key portion of a first user crypto-key associated with a first user to encrypt said second temporary key portion to form a first encrypted message;

direct issuance of said first encrypted message to the trusted third party;

apply said first temporary key portion to encrypt a first communication to form a first encrypted communication or to decrypt a second communication, including a communication encrypted with said second temporary key portion, to obtain said second communication; and  
direct issuance of said first encrypted communication to a second user;

wherein, said second temporary key portion is obtainable by applying the public key portion of the first user crypto-key to decrypt a second encrypted message, which includes the first encrypted message encrypted with a second private key portion of the first user crypto-key.

18. A programmed computer for securing communications between users of a system having a plurality of system users, each said user having an associated asymmetric crypto-key with a public key portion and a corresponding private key portion, each public key portion being accessible to the plurality of system users, each private key portion having a first private key portion known only to the associated user and a corresponding second private key portion known only to a trusted third party, comprising:

a processor (i) for generating a temporary asymmetric crypto-key having a first temporary key portion and an associated second temporary key portion, (ii) for applying the first private key portion of a first user crypto-key associated with a first user to encrypt said second temporary key portion to form a first encrypted message, (iii) for directing issuance of said first encrypted message to the trusted third party, (iv) for applying said first temporary key portion to encrypt a first communication to form a first encrypted communication or to decrypt a second communication, including a communication encrypted with said second temporary key portion, to obtain the second communication, and (v) for directing issuance of said first encrypted communication to a second user; and  
storage medium for storing said first temporary key portion;

wherein, said second temporary key portion is obtainable by applying the public key portion of the first user crypto-key to decrypt a second encrypted message, which includes the first encrypted message encrypted with a second private key portion of the first user crypto-key.

19. An article of manufacture for authenticating users of a system having a plurality of system users, each said user having an associated asymmetric crypto-key with a public key portion and a corresponding private key portion, each public key portion being accessible to the plurality of system users, each private key portion having a first private key portion known only to the associated user and a corresponding second private key portion known only to a third party, comprising:

computer readable storage medium; and

computer programming stored on said storage medium;

wherein said stored computer programming is configured to be readable from said computer readable storage medium by a computer and thereby cause said computer to operate so as to:

apply the second private key portion of a first user crypto-key associated with a first user to decrypt a first encrypted message, which includes a first message encrypted with the first private key portion of the first user crypto-key, and thereby authenticate the first user to the third party;

encrypt the first encrypted message with the second private key portion of the first user crypto-key to form a second encrypted message; and

direct issuance of the second encrypted message to the first user, wherein the first message is obtainable by application of the public key portion of the first user crypto-key to decrypt the second encrypted message and thereby authenticate the third party to the first user.

20. A article of manufacture for authenticating users of a system according to claim 19, wherein said third party is an authentication server.

21. A programmed computer for authenticating users of a system having a plurality of system users, each said user having an associated asymmetric crypto-key with a public key portion and a corresponding private key portion, each public key portion being accessible to the plurality of system users, each private key portion having a first private key portion known only to the associated user and a corresponding second private key portion known only to a third party, comprising:

a storage medium for storing the second private key portion of a first user crypto-key associated with a first user; and

a processor (i) for applying the second private key portion of the first user crypto-key to decrypt a first encrypted message, which includes a first message encrypted with the first private key portion of the first user crypto-key, and thereby authenticate the first user to the third party, (ii) for encrypting the first encrypted message with the second private key portion of the first user crypto-key to form a second encrypted message, and (iii) for directing issuance of the second encrypted message to the first user, wherein the first message is obtainable by application of the public key portion of the first user crypto-key to decrypt the second encrypted message and thereby authenticate the third party to the first user.

22. An article of manufacture for authenticating users of a system having a plurality of system users, each said user having an associated asymmetric crypto-key with a public key portion and a corresponding private key portion, each public key portion being accessible to the plurality of system users, each private key portion having a first private key portion known only to the associated user and a corresponding second private key portion known only to a third party, comprising:

computer readable storage medium; and

computer programming stored on said storage medium;

wherein said stored computer programming is configured to be readable from said computer readable storage medium by a computer and thereby cause said computer to operate so as to:

apply the second private key portion of a first user crypto-key associated with a first user to encrypt a first encrypted message, which includes a first message encrypted with a first private key portion of a first user crypto-key, to form a second encrypted message; and

direct issuance of said second encrypted message to a second user, wherein the first message is obtainable by applying the public key portion of the first user crypto-key to decrypt the second encrypted message and thereby authenticate the first user to the second user.

23. An article of manufacture for authenticating users of a system according to claim 21, wherein by obtaining said first message the signature of both the first user and the third party on the first message is verified.

24. An article of manufacture for authenticating users of a system according to claim 21, wherein said third party is an authentication server.

25. A programmed computer for authenticating users of a system having a plurality of system users, each said user having an associated asymmetric crypto-key with a public key portion and a corresponding private key portion, each public key portion being accessible to the plurality of system users, each private key portion having a first private key portion known only to the associated user and a corresponding second private key portion known only to a third party, comprising:

storage medium for storing the second private key portion of a first user crypto-key associated with a first user; and

a processor for applying the second private key portion of the first user crypto-key to encrypt a first encrypted message, which includes a first message encrypted with the first private key portion of the first user crypto-key, to form a second encrypted message, and for issuing said second encrypted message to a second user, wherein the first message is obtainable by applying the public key portion of the first user crypto-key to decrypt the second encrypted message and thereby authenticate the first user to the second user.

26. A programmed computer according to claim 24, wherein the application of the second private key portion of the first user crypto-key to encrypt the first encrypted message serves as a signature of the third party on said first message.

27. A programmed computer according to claim 24, wherein said third party is an authentication server.

\* \* \* \* \*